INTERNATIONAL CENTER FOR NUMERICAL METHODS IN ENGINEERING

A Comparison of Two Objective Stress Rates in Object-Oriented Codes

A. Rodriguez-Ferran
A. Huerta

Monograph CIMNE Nº-26, December 1994

A Comparison of Two Objective Stress Rates in Object-Oriented Codes

A. Rodriguez-Ferran and
A. Huerta

Monograph CIMNE Nº-26, December 1994

International Center for Numerical Methods in Engineering
Gran Capitán s/n, 08034 Barcelona, Spain

Diseño de la cubierta: Jordi Pa

Primera Edición, Enero 1995

Edita:
Centro Internacional de Métodos Numéricos en Ingeniería
Edificio C1, Campus Norte UPC
Gran Capitán, s/n
08034 Barcelona, España

ISBN: 84-87867-52-9
Depósito Legal: B-7370-95

A COMPARISON OF TWO OBJECTIVE STRESS RATES IN OBJECT-ORIENTED CODES

0. Introduction

1. Object-oriented codes

1.0 Introduction

1.1 An object-oriented code: Castem2000

2. Large strain solid mechanics

2.0 Introduction

2.1 Kinematics

2.2 Conservation laws

2.3 Stress tensors

2.4 Constitutive equations

2.5 Objectivity

2.6 Some objective stress rates

2.7 Finite element approximation

3. Numerical time-integration

3.1 Incremental/iterative implicit methods

3.2 Time-integration of the constitutive equation

3.3 Two stress update algorithms in CASTEM2000

4. Comparison of two time-integration algorithms

4.0 Introduction

4.1 Rigid rotation

4.2 Simple shear

4.3 Uniaxial extension

4.4 Extension and compression

4.5 Dilatation

4.6 Benchmark test: necking of a circular bar

5. Concluding remarks

Acknowledgements

Annex

A.1 Current time-integration algorithm (CEA) in CASTEM2000

A.2 New time-integration algorithm (BCN) in CASTEM2000

References

0. INTRODUCTION

Many processes of industrial interest, such as metal forming, may be numerically simulated in the context of nonlinear solid mechanics. The various sources of nonlinearity, both material and geometric, are so important that a classical linear elastic analysis is not valid. Material nonlinearity refers to the elastoplastic behaviour typically shown by metals. In most forming processes, moreover, the solid undergoes such large deformations that the variation in shape may not be neglected, as in a linear computation, thus resulting in geometric nonlinearity.

Nonlinear material behaviour is often described by a rate-form constitutive equation, relating some measure of the rate of deformation to a rate of stress. The choice of a proper stress rate is a key point in nonlinear solid mechanics, because the principle of objectivity should be respected: the **constitutive equation must be independent of** the observer. This is only achieved when objective quantities are employed. It can be shown that the material derivative of stress is not an objective tensor and, therefore, an alternative, objective stress rate is needed. Objectivity does not uniquely determine the stress rate; infinitely many can be defined. Some classical choices are reviewed.

The objectivity of the constitutive equations should be respected by the numerical algorithms employed for their time-integration. This requirement, often referred to as incremental objectivity, means that rigid rotations should be correctly treated by the algorithms.

Two such algorithms are presented in this work, and compared with the help of various simple deformation paths (rigid rotation, shear, uniaxial extension, extension-compression, and dilatation) and a necking analysis, a classical benchmark test in nonlinear solid mechanics.

An object-oriented code, CASTEM2000, has provided the framework for algorithmic development and numerical testing. In these codes, information is stored and manipulated as objects which represent the entities relevant to a finite element method computation. Objects are created with the help of operators. As a consequence, the task of developing new algorithms or performing numerical simulations is considerably easier than with a conventional code, because the user/programmer need not worry about information transfer.

This work is organized as follows: first, some basic aspects of object-oriented codes in general, and CASTEM2000 in particular, are commented is Section 1. Then, some fundamental notions on large strain solid mechanics, with special emphasis in objectivity, are reviewed in Section 2. The key point of the numerical time-integration, both of the momentum and the constitutive equation, is treated in Section 3, where the two algorithms compared in this work are presented. These algorithms are compared in Section 4. Finally, some concluding remarks are made in Section 5.

1. OBJECT-ORIENTED CODES

1.0 Introduction

The finite element method (FEM) is nowadays regarded as one of the most powerful and versatile techniques for the numerical solution of partial differential equations. It is a basic tool in the numerical simulation of a vast range of phenomena in the fields of science and engineering.

Conventional codes are designed to solve a certain range of problems, and it is not easy to modify them to extend their applicability. The main cause is the structure of the codes. They are frequently written in FORTRAN or in any other compiled language. The use of subroutines allows for a certain modularity of the code, in exchange of a careful definition of information transfer between the different modules. This process is tedious because only rudimentary objects (data structures) are employed: scalars, arrays,... The various quantities relevant in a FEM computation must be treated as objects of this type.

The compiled character of the programming language, so useful when executing the code, makes the development stage somehow cumbersome. To implement a new facility in the code, it is necessary to write the corresponding module, to compile it and to re-link the whole code again. This process is often a source of errors.

In an attempt to overcome the limitations of conventional codes, the first FEM object-oriented codes (OOC) have appeared recently, [1,2,3]. The aim is a greater flexibility of the code, both from the user's and the programmer's viewpoint: the range of applicability should be as large as possible, and further modification and improvement of the code should be easy to perform.

The essential feature of the OOC is the storage and manipulation of information through objects which are of a far more complex nature than the ones encountered in conventional codes. The basic idea is that higher complexity means more information on object structure and subsequently eases the manipulation of objects.

The various objects represent the entities needed to solve the problem. In structural mechanics, the required objects are nodes, meshes, nodal fields (displacements or loads), fields-by-element (strains or stresses), stiffness and mass matrices,... In an OOC, for instance, the finite element mesh is an object in itself instead of a matrix of nodal coordinates and a connectivity matrix. In this way, the mesh is an object on its own which can be manipulated as an entity: it may be plotted, deformed by a displacement field, support a certain finite element formulation (plane stress, plane strain, plate, shell,...).

Whatever the programming language is, the global computation process is divided into a sequence of elementary processes –which are relatively simple, but, at the same time, general enough so as to be useful in different situations–. The various stages of a FEM computation (mesh generation, choice of formulation and material parameters, definition of boundary conditions, evaluation and assembly of stiffness matrices, solving of algebraic system of equations, postprocess) correspond to one or several of these elementary

processes. In each of them, the already avalaible information is adequately treated to produce new information.

These elementary processes are represented in OOC by means of **operators**. An operator manipulates one or more existing objects to create a new object. An OOC may then be thought of as a programming environment, where a meta-language consisting of sentences of the type:

```
new_object = OPERATOR old_object_1 old_object_2 ...
```

are employed. Starting from the input data, various intermediate objects are created until the desired solution -also an object; a nodal field of displacements or a field-by-element of stresses, for instance- is obtained.

Some basic features are common to all OOC, [2]. Information is encapsulated within each object, in the sense that it may only be reached by sending a message to the object; this is known as *data encapsulation*.

Objects of the same type are grouped into classes. To ease the task of defining new classes, a feature called *inheritance* is provided. It allows a class to inherit the attributes of a more general class. When defining the class of symmetric matrices, for instance, only the specific properties caused by symmetry should be addressed; all the general properties of matrices are inherited from the class of matrices.

Operators are created to act on a range of objects as wide as possible. Thus, operator ADDITION has different meanings depending whether it is applied to integers, matrices or nodal fields. This feature is called *polimorphism*.

1.2 An object-oriented code: CASTEM2000

An object-oriented code, CASTEM2000, is used as the basic research tool in this work. This code has been developed by the Comissariat à l'Énergie Atomique (C.E.A.) and includes several kinds of objects and a large number of operators for thermo-mechanical analysis of structures, [4,5,6]. Presently various European work-teams –the Department of Applied Mathematics III of the Universitat Politècnica de Catalunya among them, [7]– cooperate in its further development. The independency between operators greatly facilitates this common task.

An object is defined in CASTEM2000 by its name, its type and its value. The name of the object is chosen by the user and allows its manipulation at the meta-language level. The type or class of an object refers to a certain data structure: mesh, nodal field, stiffness matrix,... The value of an object enables the access to the information it contains by means of a pointer. An operator is identified by its name a some sintactic rules, specyfing the number and types of objects needed as arguments.

The meta-language, called GIBIANE, is of an interactive nature: once the execution is started, CASTEM2000 is ready to accept GIBIANE sentences interactively. It is also possible to work in batch mode: several GIBIANE sentences may be included in a file and submitted to the code as a whole. Typically the two modes are combined in the development stage: the "good", tested sentences are kept in a file, while different further options are tried interactively.

An example of how a problem may be solved with CASTEM2000 is presented in Table 1, where the GIBIANE file corresponding to a simple 2-D elastic analysis is shown.

PLANE STRESS ANALYSIS OF A STEEL SHEET

```
* Setting of general options: 2-D problem, plane stress analysis,
* bilinear quadrilateral elements, graphics in X-Windows
*
 OPTION DIMENSION 2 MODE PLAN CONTRAINTE ELEMENT QUA4 TRACE X ;
*
* Mesh generation
* 1) Definition of the vertexes p1, p2, p3 and p4 of the sheet
* 2) Definition, via operator DROITE, of lines l1, l2, l3 and l4
* 3) Mesh generation with operator DALLER
* 4) Mesh display with operator TRACE
*
 p1 = 0.  0.  ; p2 = 1.  0.  ; p3 = 1.  0.5 ; p4 = 0.  0.5 ;
 l1 = p1 DROITE 5 p2 ; l2 = p2 DROITE 5 p3 ;
 l3 = p3 DROITE 5 p4 ; l4 = p4 DROITE 5 p1 ;
 sup1 = DALLER l1 l2 l3 l4 ;
 TRACER sup1 ;
*
* Model and material
* 1) Definition of mechanical model with operator MODL
* 2) Definition of material parameters with operator MATR
*
 mod1 = MODL sup1 MECANIQUE ELASTIQUE ISOTROPE QUA4 ;
 mat1 = MATR mod1 YOUNG 2.1e7 NU 0.3 RHO 7.85 ;
*
* Boundary conditions
* 1) Concentrated force in point p3 via operator FORCE
* 2) Prescribed displacements in side l1 via operator BLOQUER
*
 load3 = FORCE FX 3000.  FY 3000.  p3 ;
 bc1 = BLOQUER DEPLACEMENT l1 ;
*
* Stiffness matrix
* 1) Internal stiffness matrix with operator RIGIDITE
* 2) Assembly of internal and boundary stiffnesses with operator ET
*
 stiff1 = RIGIDITE mod1 mat1 ;
 stiff2 = stiff1 ET bc1 ;
*
* Solver
* 1) Solution of linear system with operator RESOUDRE
*
 disp1 = RESOUDRE stiff2 load3 ;
*
* Postprocess and results
* 1) Stress with operator SIGMA
* 2) Von Mises stress with operator VMISES
* 3) Display with operator TRACER
*
 tens1 = SIGMA mod1 mat1 disp1 ;
 vmis1 = VMISES mod1 tens1 ;
 TRACER sup1 mod1 vmis1 ;
```

Table 1: CIBLANE file for a 2-D elastic analysis

2. LARGE STRAIN SOLID MECHANICS

2.0 Introduction

In this section some basic notions in large strain solid mechanics are reviewed. First the kinematics and the various alternative measures of strain are presented. Then the conservation laws that govern the problem are shown followed by the different stress tensors that appear in a large deformation analysis. The nonlinear constitutive equations to account for geometric and material nonlinearity are developed, after discussing the principle of objectivity that these equations must comply with. Then various classical objective stress rates are presented and, finally, the semi-discretized equations resulting from finite element approximation are developed.

2.1 Kinematics

The starting point in continuum mechanics is the description of the kinematics of the body. The equation of motion η relates the material domain R_X, where particles are identified by their material coordinates X, to the spatial domain R_x, where points are identified by their spatial coordinates x:

$$\begin{aligned} \eta\colon\ R_X \times [\,0,\infty) &\longrightarrow R_x \times [\,0,\infty) \\ (X,t) &\longmapsto (x,t) \end{aligned} \tag{1a}$$

with

$$x_i = x_i(X,t) \quad ; \quad t = t \tag{1b}$$

Equation (1b) explicitly states the particular nature of η : *i)* the spatial coordinates x depend both on the material particle X and time t, and *ii)* physical time is measured by the same variable t in both material and spatial domains. For every fixed instant t, the mapping:

$$\begin{aligned} \eta_t\colon\ R_X &\longrightarrow R_x \\ X &\longmapsto x \end{aligned} \tag{1c}$$

defines the configuration Ω_t of the body at time t, see Figure 1.

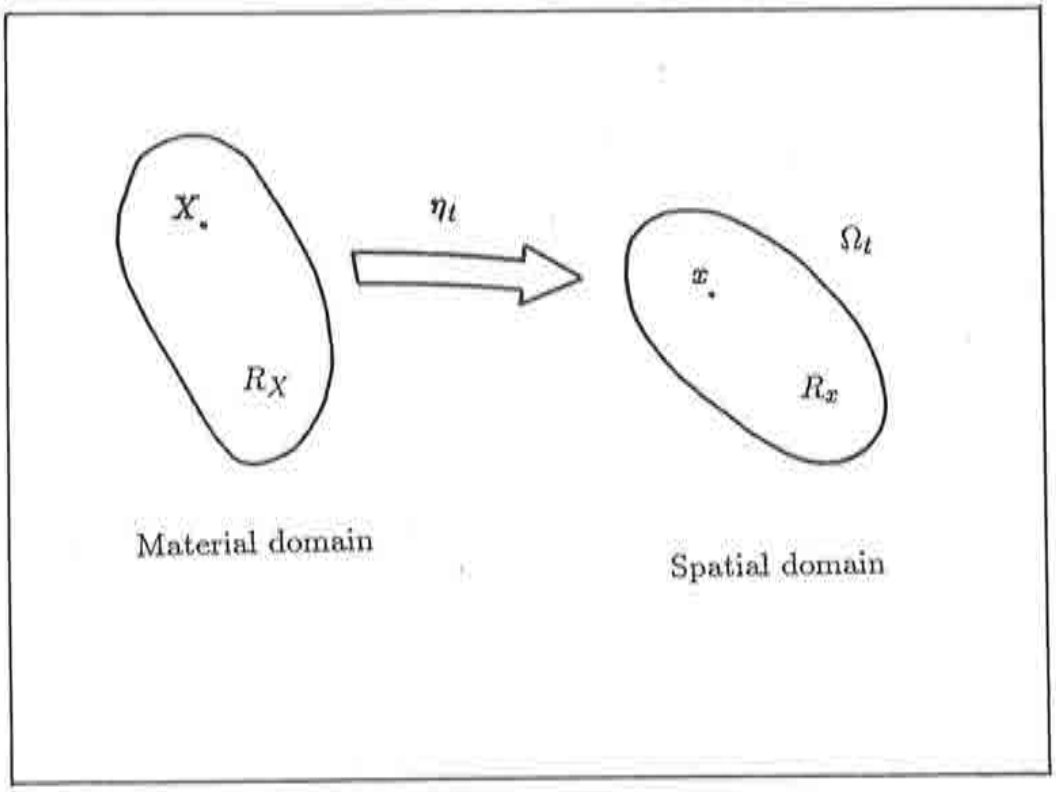


Figure 1: Material and spatial domains at time t.

The initial, undeformed configuration Ω_0 is often taken as a reference. The material coordinates are then simply the spatial coordinates at time t_0, and the material displacements are defined as:

$$\boldsymbol{u}(t) = \boldsymbol{x}(t) - \boldsymbol{X} \tag{2}$$

Once the material displacements are defined, the kinematical description continues with strain representation. The fundamental tensor is the deformation gradient $\boldsymbol{F}$,

$$\boldsymbol{F} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} \tag{3}$$

which admits the polar decomposition:

$$\boldsymbol{F} = \boldsymbol{R} \cdot \boldsymbol{U} = \boldsymbol{V} \cdot \boldsymbol{R} \tag{4}$$

where $\boldsymbol{R}$, the rotation tensor, is orthogonal, and $\boldsymbol{U}$, $\boldsymbol{R}$, both symmetric and positive-definite, are called the the right and left stretch tensor respectively.

Various strain tensors may be defined by means $\boldsymbol{F}$. The Lagrange strain tensor, for instance, is:

$$\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{F}^T \cdot \boldsymbol{F} - \boldsymbol{I}\right) \tag{5}$$

Another tensor representing strain is the spatial gradient of velocity $\boldsymbol{l}$:

$$\boldsymbol{l} = \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}} \tag{6}$$

which is related to the deformation gradient $\boldsymbol{F}$ by:

$$\dot{\boldsymbol{F}} = \left.\frac{\partial \boldsymbol{F}}{\partial t}\right|_{\boldsymbol{X}} = \boldsymbol{l} \cdot \boldsymbol{F} \tag{7}$$

where both $\dot{\cdot}$ and $\left.\frac{\partial}{\partial t}\right|_{\boldsymbol{X}}$ mean material time derivative (i.e, holding the particle $\boldsymbol{X}$ fixed), and yields relevant tensors if decomposed into symmetric and skew-symmetric parts. The

symmetric part is the rate-of-deformation tensor:

$$\boldsymbol{d} = \frac{1}{2}\left(\boldsymbol{l} + \boldsymbol{l}^T\right) \tag{8}$$

while the skew-symmetric part is the spin rate tensor:

$$\boldsymbol{\omega} = \frac{1}{2}\left(\boldsymbol{l} - \boldsymbol{l}^T\right) \tag{9}$$

Note that rigid rotations and "pure" deformations contribute to the deformation gradient $\boldsymbol{F}$, see Eq. (4). In a similar manner, the decomposition of $\boldsymbol{l}$ also separates these two components of motion: $\boldsymbol{d}$ describes "pure" deformation, while $\boldsymbol{\omega}$ takes care of rigid rotations.

An alternative measure of rotations is the rate of rotation $\boldsymbol{\Omega}$, related to the rotation tensor $\boldsymbol{R}$ by:

$$\boldsymbol{\Omega} = \dot{\boldsymbol{R}} \cdot \boldsymbol{R}^T \tag{10}$$

A very common simplification in solid mechanics is that of small deformations. If displacements, rotations and strains are small enough, two important points follow: *i)* the relation between displacements and strain is linear (because second-order terms such as those present in (5) are negligible) and *ii)* the configuration Ω_t at any instant is so similar to the initial configuration Ω_0 as to be confounded with it. This enables the use of a constant configuration Ω_0 to solve the governing equations throughout the analysis, with no need to update the various magnitudes to the current configuration. For this reason, a *geometrically linear* problem results. The small strain hypothesis is typical, for instance, in structural mechanics, [8].

In some other problems, on the contrary, displacements and strains are large when compared to the initial dimensions of the body. Metal forming processes [9,10] are a clear example of that, see Figure 2. It is no longer possible to disregard second-order terms or –and this is the fundamental point– to confound configurations Ω_0 and Ω_t. When solving the governing equations, Ω_t should be taken as the domain. As the motion $\boldsymbol{\eta}$ that transforms Ω_0 into Ω_t is precisely one of the unknowns, a *geometrically nonlinear* problem is obtained.

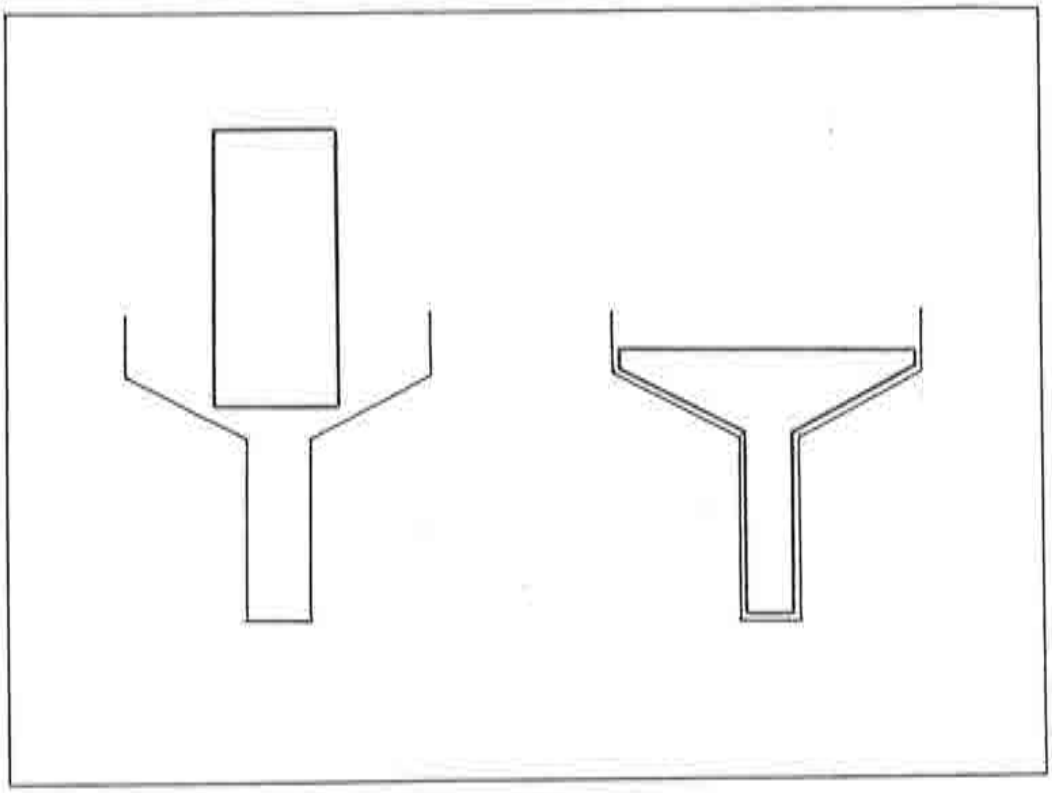


Figure 2: Large strains in a metal forming process.

2.2 Conservation laws

The fundamental equations in continuum mechanics state the conservation of mass, momentum and energy:

Mass:

$$\left.\frac{\partial \rho}{\partial t}\right|_X = -\rho \frac{\partial v_j}{\partial x_j} \tag{11}$$

Momentum:

$$\rho \left.\frac{\partial v_i}{\partial t}\right|_X = \frac{\partial \sigma_{ij}}{\partial x_j} + b_i \tag{12}$$

Energy:

$$\rho \left.\frac{\partial e}{\partial t}\right|_X = \sigma_{ij} \frac{\partial v_i}{\partial x_j} + \rho a - \frac{\partial q_i}{\partial x_i} \tag{13}$$

where ρ is the density, $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{b}$ is the force per unit volume, e is the especific internal energy, a is the internal generation of heat and $\boldsymbol{q}$ is the heat flux.

If mechanical and thermal effects are uncoupled, Eqs. (11) and (12) are solved independently from Eq. (13). Two assumptions are usual when studying a certain range of solid mechanics problems, including some metal forming processes: constant density and quasistatic process. The inertia forces $\rho \left.\frac{\partial v_i}{\partial t}\right|_X$ in Eq. (12) are neglected, and the momentum balance becomes a static equilibrium equation:

$$\frac{\partial \sigma_{ij}}{\partial x_j} + b_i = 0 \tag{14}$$

It is very important to note that a process being quasistatic does not imply null accelerations and therefore constant material velocities. It is only *inertia forces*, not accelerations, that are disregarded in comparison with stress gradients and/or body forces. Some forming processes with relevant accelerations, such as forging, may be studied as quasistatic processes because variations of stresses within the piece are much larger.

Boundary conditions are needed to complement the conservation laws. The boundary Γ of Ω_t is assumed to be composed of two parts Γ^g and Γ^h with prescribed velocities and tractions respectively:

$$v_i = g_i \qquad \text{in} \quad \Gamma^g \tag{15a}$$

$$\sigma_{ij} n_j = h_i \qquad \text{in} \quad \Gamma^h \tag{15b}$$

where $\boldsymbol{g}$ are prescribed velocities, $\boldsymbol{h}$ prescribed tractions and $\boldsymbol{n}$ is the outward unit normal to Γ.

2.3 Stress tensors

It has been shown in Section 2.1 that various tensors may be chosen to represent strains; a similar situation occurs with stresses.

The most common choice is the Cauchy stress tensor $\boldsymbol{\sigma}$, defined in the current configuration Ω_t and already presented in Eq. (12). This tensor has a clear physical meaning, because it involves only forces and surfaces in the current configuration. Experimental stress measures taken in a laboratory correspond to Cauchy stresses. This tensor is also known as the true stress tensor.

In a large strain context, other representations of stress are possible and indeed useful. The key idea, [11], is that Ω_0 and Ω_t are different configurations, so tensor fields defined in each configuration cannot be combined by operations such as substraction and addition. Let ${}^0\boldsymbol{\sigma}$ and ${}^t\boldsymbol{\sigma}$ be the Cauchy stress tensors at Ω_0 and Ω_t respectively; the increment of stress may not be defined as ${}^t\boldsymbol{\sigma} - {}^0\boldsymbol{\sigma}$, because the two tensors are referred to different configurations. As stress increments will be needed to update stresses, a proper definition is required.

An alternative representation of stress is the second Piola-Kirchhoff tensor $\boldsymbol{S}$, defined as the pull-back of $\boldsymbol{\sigma}$:

$$S = JF^{-1} \cdot \boldsymbol{\sigma} \cdot F^{-T} \tag{16}$$

where $J = \det \boldsymbol{F}$ is the Jacobian, and the inverse of the deformation gradient $\boldsymbol{F}^{-1}$ is employed to transform $\boldsymbol{\sigma}$ from Ω_t to Ω_0, see Figure 3a. The Jacobian J reflects the variation in size of an element of area and does not appear in the standard definition of pull-back in differential geometry, [12]. Because of this, Eq. (16) is called the (pull-back) Piola transformation.

It is very important to note that S represents the state of stress at time t but referred to configuration Ω_0, and should not be confused with ${}^0\boldsymbol{\sigma}$, the stress at initial time t_0.

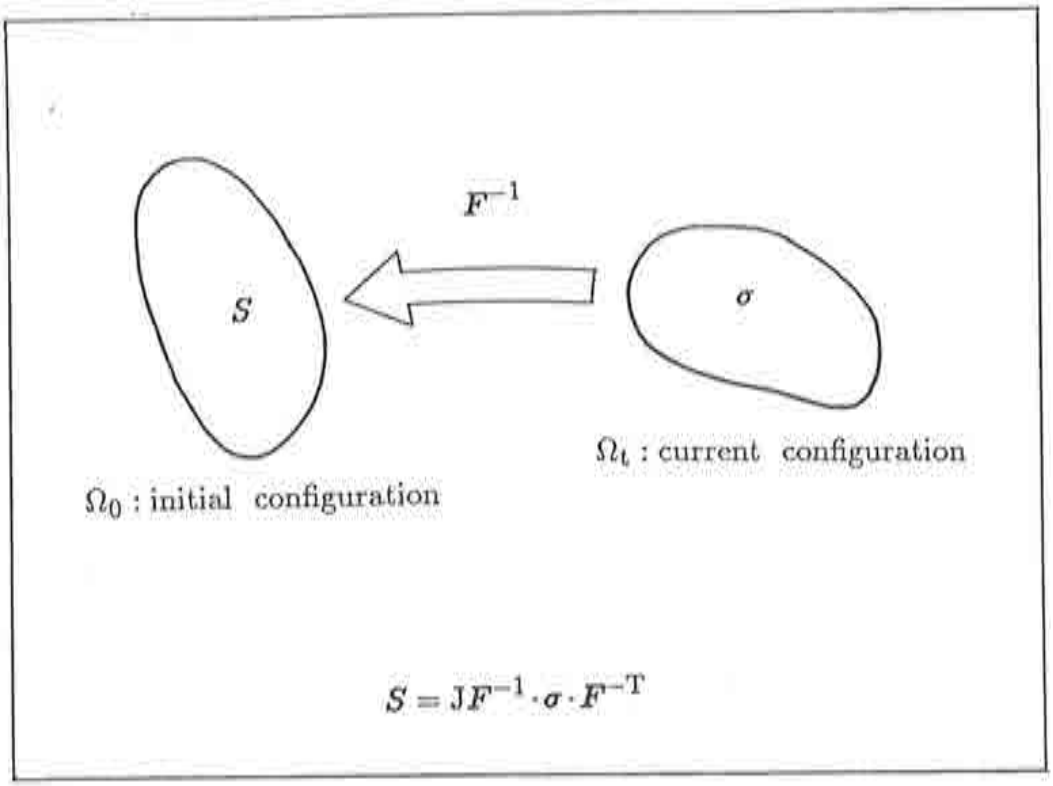


Figure 3a: Pull-back Piola transformation.

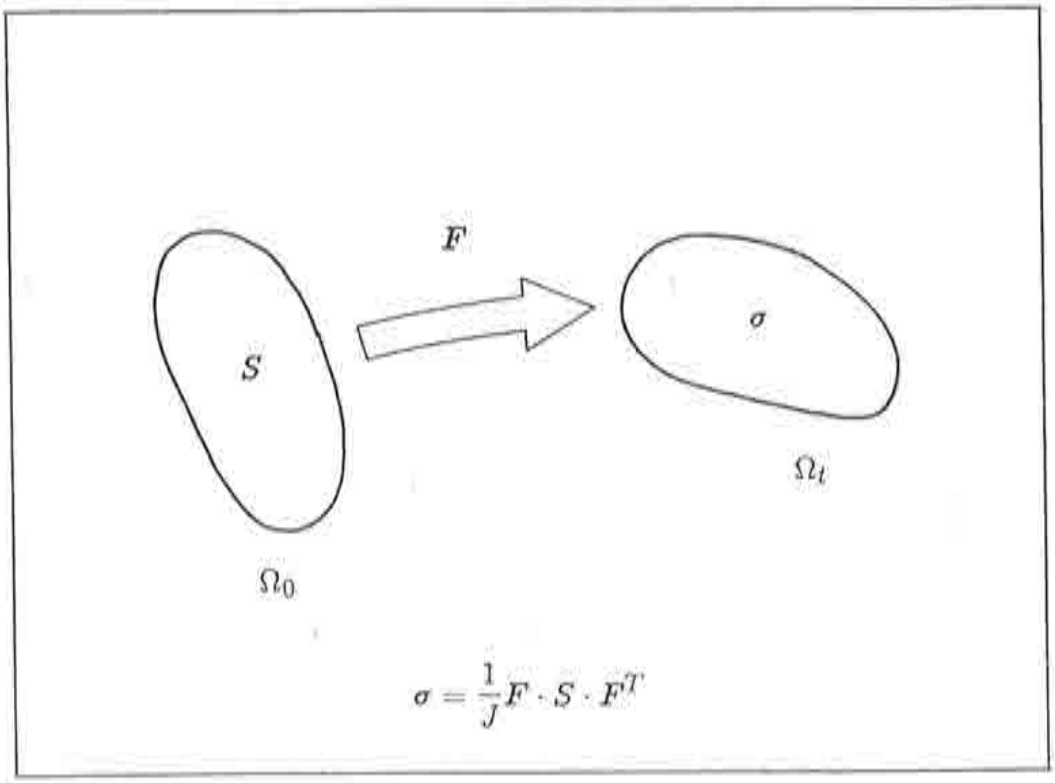


Figure 3b: Push-forward Piola transformation.

Equation (16) may be reversed, and then $\boldsymbol{\sigma}$ may be seen as the push-forward of $\boldsymbol{S}$:

$$\boldsymbol{\sigma} = \frac{1}{J}\boldsymbol{F} \cdot \boldsymbol{S} \cdot \boldsymbol{F}^T \tag{17}$$

where $\boldsymbol{F}$ transforms $\boldsymbol{S}$ from Ω_0 to Ω_t. As before, (17) is not a "pure" push forward but rather a (push-forward) Piola transformation, see Figure 3b.

With the help of Eqs. (16) and (17), the stress increment may be represented as

$${}^t\Delta\boldsymbol{\sigma} = {}^t\boldsymbol{\sigma} - J^{-1}\boldsymbol{F} \cdot {}^0\boldsymbol{\sigma} \cdot \boldsymbol{F}^T \tag{18a}$$

or

$${}^0\Delta\boldsymbol{\sigma} = J\boldsymbol{F}^{-1} \cdot {}^t\boldsymbol{\sigma} \cdot \boldsymbol{F}^{-T} - {}^0\boldsymbol{\sigma} \tag{18b}$$

referred to Ω_t or Ω_0 respectively. The Piola transformations are employed to refer the two tensors to a common configuration, where the substraction can be properly performed.

2.4 Constitutive equations

Two equations of state are needed to completely define the initial boundary value problem represented by Eqs. (11), (12), (13) and (15). The first one relates temperature θ and density to internal energy:

$$e = e(\rho, \theta) \tag{19}$$

The second one relates stress and/or its derivatives to velocity and/or its derivatives, temperature and density. Most of such equations employed in continuum mechanics are of one of the two following general forms:

$$\text{Type 1}: \quad \boldsymbol{\sigma} = \boldsymbol{C}_1(\theta, \rho, \boldsymbol{v}) \tag{20a}$$

$$\text{Type 2}: \quad \left.\frac{\partial \boldsymbol{\sigma}}{\partial t}\right|_X = \boldsymbol{C}_2(\theta, \rho, \boldsymbol{v}, \boldsymbol{\sigma}) \tag{20b}$$

Equation (20a) represents a "no memory" material, in the sense that there is no dependence on stress history. The stress at any given instant can be determined from other instantaneous fields $(\theta, \rho, \boldsymbol{v})$ at that same instant, and does not depend on the previous state of the material.

Equation (20b), on the other hand, models a material with "memory". The constitutive behaviour of the continuum is affected by its current state of stress, and thus a rate form equation, providing material derivatives of stress, results. It is no longer possible to compute stresses directly from the instantaneous state of the material; they must be updated by integrating Eq. (20b) in time. Several nonlinear solid materials may be modeled by such type of equations. For hypoelastic materials, for instance, $\boldsymbol{C}_2(\theta, \rho, \boldsymbol{v}, \boldsymbol{\sigma})$ is assumed to depend linearly on the rate-of-deformation tensor:

$$\boldsymbol{C}_2(\theta, \rho, \boldsymbol{v}, \boldsymbol{\sigma}) = \boldsymbol{C} : \boldsymbol{d} \tag{21}$$

where $\boldsymbol{C}$ is the modulus tensor. Equation (21) can be extended, by means of and additive decomposition of $\boldsymbol{d}$ into elastic and plastic parts, to account for elastoplastic behaviour, [17].

2.5 Objectivity

It has been already mentioned that, since configurations at various instants are too different apart, tensors must be referred to a common configuration before they can be added or substracted.

There is an additional requirement regarding the constitutive equations: the principle of objectivity, [13]. If the constitutive equations really describe the physical behaviour of the continuum, they must be independent of the observer. In other words, they must remain invariant under a time-dependent change of reference frame.

This requirement is fulfilled if objective quantities appear in constitutive equations. A quantity is said to be objective if it transforms in a proper tensorial manner under a superposed rigid-body motion. Let the rigid rotation be represented by an orthogonal rotation tensor $\boldsymbol{Q}$, $(\boldsymbol{Q}^{-1} = \boldsymbol{Q}^T)$ and a translation $\boldsymbol{a}$. The time-dependent relation between old and new coordinates is then:

$$x^{\text{new}}(t) = \boldsymbol{Q}(t)x + a(t) \tag{22}$$

It is postulated that the Cauchy stress tensor $\boldsymbol{\sigma}$ is objective. As it is a second-order tensor, it transforms according to:

$$\boldsymbol{\sigma}^{\text{new}}(t) = \boldsymbol{Q}(t) \cdot \boldsymbol{\sigma}(t) \cdot \boldsymbol{Q}(t)^T \tag{23}$$

where compact notation is employed: $\boldsymbol{\sigma}(t) \equiv \boldsymbol{\sigma}(\boldsymbol{x}, t)$ and $\boldsymbol{\sigma}^{\text{new}}(t) \equiv \boldsymbol{\sigma}^{\text{new}}(\boldsymbol{x}, t)$.

Derivating Eq. (23) with respect to time shows that the material derivative of an objective tensor is not objective:

$$\dot{\sigma}^{\text{new}} = \dot{Q} \cdot \sigma \cdot Q^T + Q \cdot \dot{\sigma} \cdot Q^T + Q \cdot \sigma \cdot \dot{Q}^T \neq Q \cdot \dot{\sigma} \cdot Q^T \tag{24}$$

This invalidates the use of $\dot{\sigma}$ as the stress rate in a rate-form constitutive equation, Eq. (20b). An alternative, objective stress rate σ^* is therefore needed. The choice is by no means unique; some options will be shown in Section 2.6.

As for the rate-of-deformation tensor d, it can be shown that it is an objective tensor, so it may be employed to represent strains in a constitutive equation. Indeed, the hypoelastic constitutive equation is rewritten, in a large strain framework, as:

$$\sigma^* = C : d \tag{25}$$

The non-objectivity of $\dot{\sigma}$ implies that σ^* must be used in Eq. (25).

In fact, the law $\dot{\sigma} = C : d$, valid for small strain analysis, provides unrealistic stress distributions in very simple large strain tests.

Consider, for example, the rigid body rotation problem depicted in Figure 4a. The rectangle $ABCD$ is initially subjected to a uniaxial stress σ^o_{xx} and rotates around vertex A with a constant angular velocity ω. The velocity field is:

$$(v_x, v_y) = (\omega y, -\omega x) \tag{26}$$

and the spatial gradient of velocity l is a skew-symmetric tensor, thus coincident with the spin rate ω:

$$l = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix} = \omega \tag{27}$$

As expected, the rate-of-deformation d is a null tensor in this rigid-body, strain-free motion. If $\dot{\sigma}$ is selected as stress rate in Eq. (25):

$$d = 0 \quad \Rightarrow \quad \dot{\sigma} = C : d = 0 \tag{28}$$

Since $\dot{\sigma} = 0$, the stress distribution is constantly equal to the initial one and does not follow the rotation. After a 90° turn, for example, the stress of Figure 4b is predicted, where the uniaxial stress is now through the short dimension of the rectangle $ABCD$. This

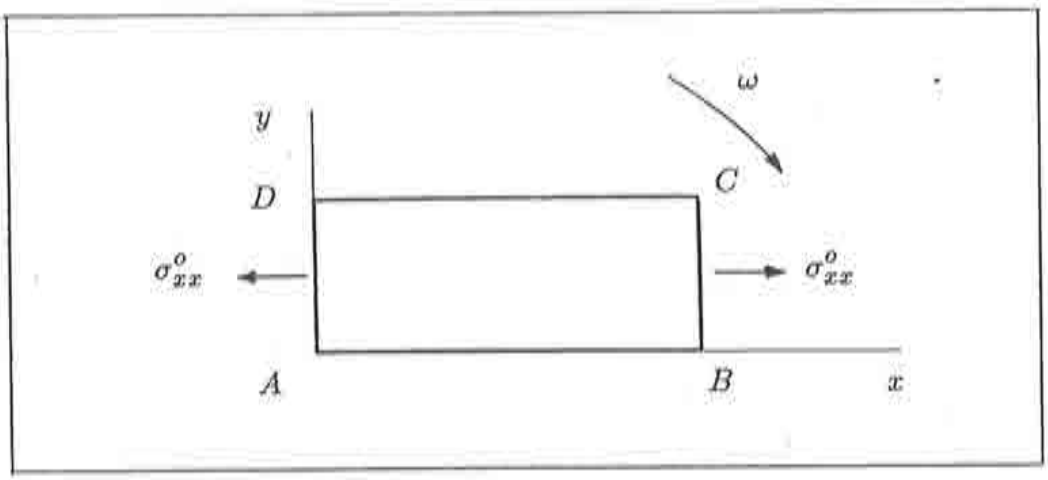


Figure 4a: Initial stress state.

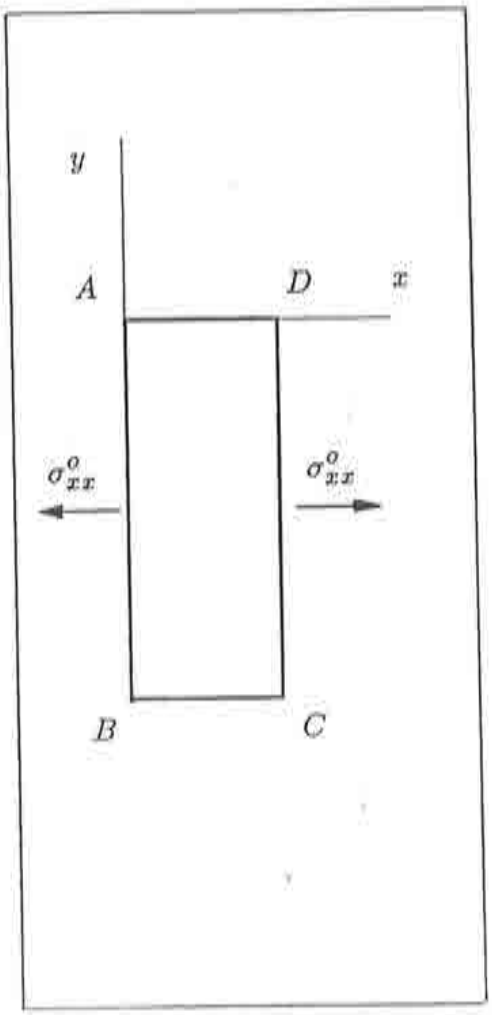


Figure 4b: Predicted stress with material stress rate $\dot{\sigma}$.

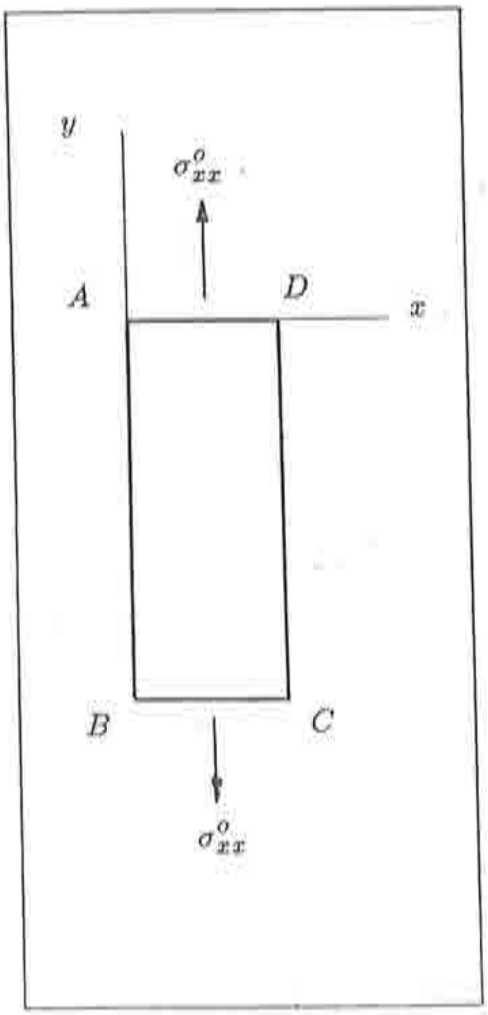


Figure 4c: Predicted stress with objective stress rate σ^*.

result is fully unsatisfactory; a simple rotation of the initial stress should be obtained, see Figure 4c. This correct answer can be achieved by using an objective stress rate (see Section 4.1).

2.6 Some objective stress rates

The requirement that the stress rate be objective does not uniquely determine it. Some options reviewed in [11] are the Jaumann rate,

$$\sigma^*_J = \dot{\sigma} + \sigma \cdot \omega - \omega \cdot \sigma \tag{29a}$$

the Green-Naghdi rate,

$$\sigma^*_{GN} = \dot{\sigma} + \sigma \cdot \Omega - \Omega \cdot \sigma \tag{29b}$$

and the Truesdell rate,

$$\sigma^*_T = \dot{\sigma} - l \cdot \sigma - \sigma \cdot l^T + \mathrm{tr}(d)\sigma \tag{29c}$$

where *tr* denotes *trace*.

It can be easily checked that the terms in the RHS of Eq. (29) additional to the material rate $\dot{\sigma}$ ensure that the defined rates are indeed objective. Either by means of the spin rate ω, the rate of rotation Ω or the gradient of velocity l, the non-objectivity of $\dot{\sigma}$ is compensated, and an objective σ^* is obtained.

Regarding the Truesdell rate, it has been defined in Eq. (29c) in terms of the *Eulerian* tensors σ, l and d, referred to the current configuration. An alternative expression, which provides insight into its physical meaning and is useful from an algorithmic viewpoint, is:

$$\sigma^*_T = \frac{1}{J} F \cdot \dot{S} \cdot F^T \tag{30}$$

The Truesdell rate is then seen as the push-forward Piola transformation of the material derivative of the second Piola-Kirchhoff stress tensor S. Instead of derivating with respect to time the Cauchy stress tensor directly, which yields the non-objective material rate $\dot{\sigma}$, the Truesdell rate proceeds in three steps: *i)* σ is pulled-back into S, *ii)* the material derivative of S is performed and *iii)* the resulting rate is pushed-forward into the current configuration. The key point is that the material derivative of a material tensor (i.e., a tensor referred to the initial configuration) yields an objective tensor.

2.7 Finite element approximation

For a wide range of problems in solid mechanics, it is a common assumption that *i)* mechanical and thermal effects are uncoupled and *ii)* the density is a constant. The attention is then focused on momentum balance, Eq. (12).

The weak form of Eq. (12) is obtained by multiplying by test functions δv_i and employing the divergence theorem to account for the traction force on the boundary Γ^h:

$$\int_\Omega \rho \delta v_i \frac{\partial v_i}{\partial t} d\Omega + \int_\Omega \rho \delta v_i v_j \frac{\partial v_i}{\partial x_j} d\Omega + \int_\Omega \frac{\partial \delta v_i}{\partial x_j} \sigma_{ij} d\Omega = \int_\Omega \delta v_i b_i d\Omega + \int_{\Gamma^h} \delta v_i h_i d\Gamma \qquad (31)$$

A finite element method proceeds by dividing the domain into elements. When appropriate sets of shape and test functions are chosen to interpolate the velocity, the matrix equation corresponding to Eq. (31) is:

$$M \frac{dv}{dt} + N v + f_{\text{int}} = f_{\text{ext}} \qquad (32)$$

where M is the mass matrix, N the convective matrix, v the velocity vector, f_{int} the internal force vector and f_{ext} the external load vector. Equations (31) and (32) simply state the equilibrium of forces, including the inertia terms caused by acceleration.

A common and physically meaningful hypothesis is to neglect inertia terms, Eq. (14). The resulting weak form is then:

$$\int_\Omega \frac{\partial \delta v_i}{\partial x_j} \sigma_{ij} d\Omega = \int_\Omega \delta v_i b_i d\Omega + \int_{\Gamma^h} \delta v_i h_i d\Gamma \qquad (33)$$

with the following FEM equation:

$$f_{\text{int}} = f_{\text{ext}} \qquad (34)$$

which models static processes. Equation (34) is typical of classical structural analysis, [8].

3. NUMERICAL TIME-INTEGRATION

3.1 Incremental/iterative implicit methods

Due to the semi-discretization (in space) associated to the FEM, the partial differential equations of momentum balance becomes a system of ordinary differential equations to be

solved by numerical time integration, in conjunction with the constitutive equation. If inertia effects are present, the two equations to solve are Eqs. (32) and (20b):

Transient analysis

$$M\frac{dv}{dt} + Nv + f_{\text{int}} = f_{\text{ext}} \tag{A}$$

$$\frac{\partial \sigma}{\partial t}\Big|_X = C(\theta, \rho, v, \sigma)$$

where t is physical time, and time integration yields the transient behaviour of the continuum. If, on the other hand, a quasistatic analysis is performed, Eqs. (34) and (20b) need to be solved:

Quasistatic analysis

$$f_{\text{int}} = f_{\text{ext}} \tag{B}$$

$$\frac{\partial \sigma}{\partial t}\Big|_X = C(\theta, \rho, v, \sigma)$$

and t is just a convenient time-like parameter employed to represent the progressive application of the total external load. By doing so, it is possible to account for geometric and material nonlinearities. In fact, t can also be seen as associated to the continuation method for solving the system of nonlinear equations. The definition of velocity, rate of deformation, material time derivatives..., is identical in both cases.

If an implicit time-integration method is chosen, velocities and accelerations are written, by means of difference formulas, as functions of displacements. The fundamental unknowns are then the incremental material displacements Δu from one (known) equilibrium configuration Ω_n at time t_n to a new (unknown) equilibrium configuration Ω_{n+1} at time t_{n+1}, a time increment Δt later.

Implicit methods are especially adequate to treat quasistatic processes, (B). The equilibrium of forces, (34), may be recast as:

$$r(u) = f_{\text{ext}}(u) - f_{\text{int}}(u) = 0 \tag{35}$$

where the nonlinear dependence that both external and internal forces may have on material displacements is explicitly stated. The residual forces $\boldsymbol{r}$ are zero if equilibrium at time t_{n+1} is attained.

Nonlinear systems of equations like Eq. (35) may be solved by a number of iterative techniques, [14-16]. The two key ideas are that *i)* a linearized form of Eq. (35) is used to get an initial prediction and successive corrections to $\Delta\boldsymbol{u}$ and *ii)* the constitutive equation (25) is integrated after each tentative configuration Ω_{n+1} is computed to get stresses and check equilibrium.

As stated above, the solution to Eq. (35) is built up in an incremental manner. Starting from an equilibrium configuration Ω_n (${}^n\boldsymbol{f}_{\text{ext}} = {}^n\boldsymbol{f}_{\text{int}}$), time is advanced by a time increment Δt, and a material displacement increment $\Delta\boldsymbol{u}$ is sought to get ${}^{n+1}\boldsymbol{f}_{\text{ext}} = {}^{n+1}\boldsymbol{f}_{\text{int}}$. In order to compute a first approximation of $\Delta\boldsymbol{u}$, Eq. (35) is linearized into:

$$\boldsymbol{K}^0 \Delta\boldsymbol{u}^0 = \Delta\boldsymbol{f}_{\text{ext}} \tag{36}$$

where $\boldsymbol{K}^0$ is the so-called tangent stiffness matrix and $\Delta\boldsymbol{f}_{\text{ext}}$ is the increment of external loads. The common situation where $\Delta\boldsymbol{f}_{\text{ext}}$ depends only on time and not on displacements is assumed. The basic ideas, however, can be extended to more general, deformation-dependent load conditions, such as fluid pressure acting on the contour. The right superscripts in Eq. (36) are counters that anticipate the need of iteration caused by approximation in the linearization process.

Once $\Delta\boldsymbol{u}^0$ is obtained by solving the *linear* system (36), a candidate Ω^0_{n+1} is known. By integrating the constitutive equation (25), stresses and internal forces ${}^{n+1}\boldsymbol{f}^0_{\text{int}}$ can be computed. The equilibrium is then checked in Ω^0_{n+1} and, in general, not verified:

$${}^{n+1}\boldsymbol{r}^0 = {}^{n+1}\boldsymbol{f}_{\text{ext}} - {}^{n+1}\boldsymbol{f}^0_{\text{int}} \neq \boldsymbol{0} \tag{37}$$

The approximate character of $\Delta\boldsymbol{u}^0$ induces lack of equilibrium. The off-balance forces ${}^{n+1}\boldsymbol{r}^0$ are then used to correct the displacements, according to an iterative scheme:

$$\boldsymbol{K}^i \delta\boldsymbol{u}^i = {}^{n+1}\boldsymbol{r}^i \tag{38}$$

where $\boldsymbol{K}^i$ is an iteration stiffness matrix, $\delta\boldsymbol{u}^i = \Delta\boldsymbol{u}^{i+1} - \Delta\boldsymbol{u}^i$ is the correction in incremental material displacements and ${}^{n+1}\boldsymbol{r}^i$ are residual forces: every time a candidate configuration Ω^i_{n+1} is obtained, stresses must be updated from the previous configuration Ω_n by integrating the rate equation (25). By doing so, the internal forces ${}^{n+1}\boldsymbol{f}^i_{\text{int}}$ and then the residual forces ${}^{n+1}\boldsymbol{r}^i$ may be computed. The iteration matrix $\boldsymbol{K}^i$ in Eq. (38) depends on the technique employed to solve the nonlinear system of equations, Eq. (35). It may be, for

instance, the true tangent stiffness matrix at each iteration (full Newton-Raphson method), a constant stiffness matrix throughout each increment (modified Newton-Raphson method) or a secant approximation to the tangent stiffness matrix (quasi-Newton methods), [8,14,15].

3.2 Time-integration of the constitutive equation

As remarked in [17], stress update (i.e., time-integration of the constitutive equation) is the central problem in nonlinear solid mechanics and affects fundamentally the accuracy of the overall algorithm.

The numerical algorithm employed in the stress update should preserve the objectivity of the constitutive equation (25), attained through the careful definition of objective stress rates, Eq. (29): if the solid undergoes a rigid body rotation, the algorithm should rotate stresses accordingly, with no additional, spurious, stress variations. This requirement is often referred to as incremental objectivity, [17].

In the context of the incremental build-up of the solution, it is useful to define the incremental versions of the tensors presented in Eqs. (3) and (5). Let ${}^{n}\boldsymbol{F}$ and ${}^{n+1}\boldsymbol{F}$ be the deformation gradients relating Ω_n and Ω_{n+1} respectively to the reference configuration Ω_0, see Figure 5. The incremental deformation gradient ${}^{n}\boldsymbol{\Lambda}$ is:

$$ {}^{n}\boldsymbol{\Lambda} = {}^{n+1}\boldsymbol{F} \cdot {}^{n}\boldsymbol{F}^{-1} \tag{39} $$

which refers configuration Ω_{n+1} to Ω_n.

The corresponding incremental Lagrange strain tensor is then:

$$ {}^{n}\boldsymbol{E} = \frac{1}{2}\left({}^{n}\boldsymbol{\Lambda}^{T} \cdot {}^{n}\boldsymbol{\Lambda} - \boldsymbol{I} \right) \tag{40} $$

A numerical algorithm is said to be *incrementally objective*, [17], if stresses are properly updated when the continuum undergoes a rigid body motion. Let Ω_n and Ω_{n+1} be the configurations of a rotating body at instants t_n and t_{n+1} respectively, see Figure 6. The incremental deformation gradient ${}^{n}\boldsymbol{\Lambda}$ is then an orthogonal tensor ${}^{n}\boldsymbol{R}$ representing a rigid rotation. The numerical algorithm is then required to predict a stress state at t_{n+1} that is simply a rotation of the stress state at t_n:

$$ {}^{n+1}\boldsymbol{\sigma} = {}^{n}\boldsymbol{R} \cdot {}^{n}\boldsymbol{\sigma} \cdot {}^{n}\boldsymbol{R}^{T} \tag{41} $$

Incremental objectivity is commonly presented as the discrete counterpart of the principle of objectivity. However, and in agreement with [18], the authors dislike

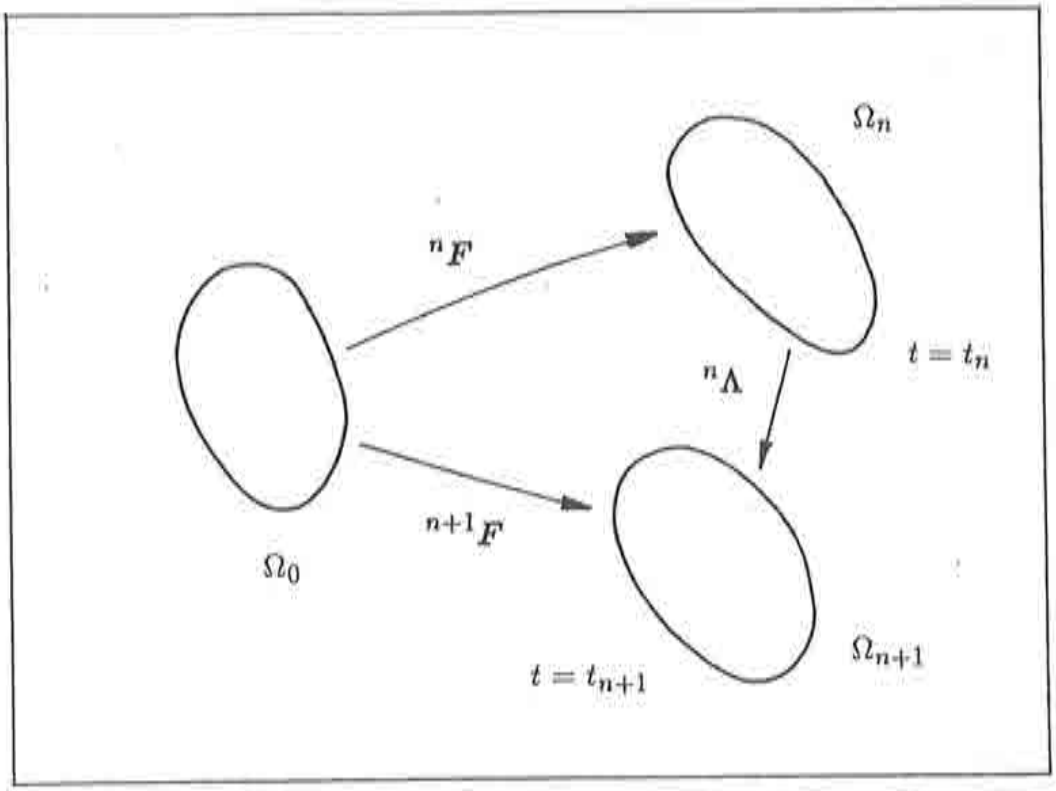


Figure 5: Deformation gradients in incremental analysis.

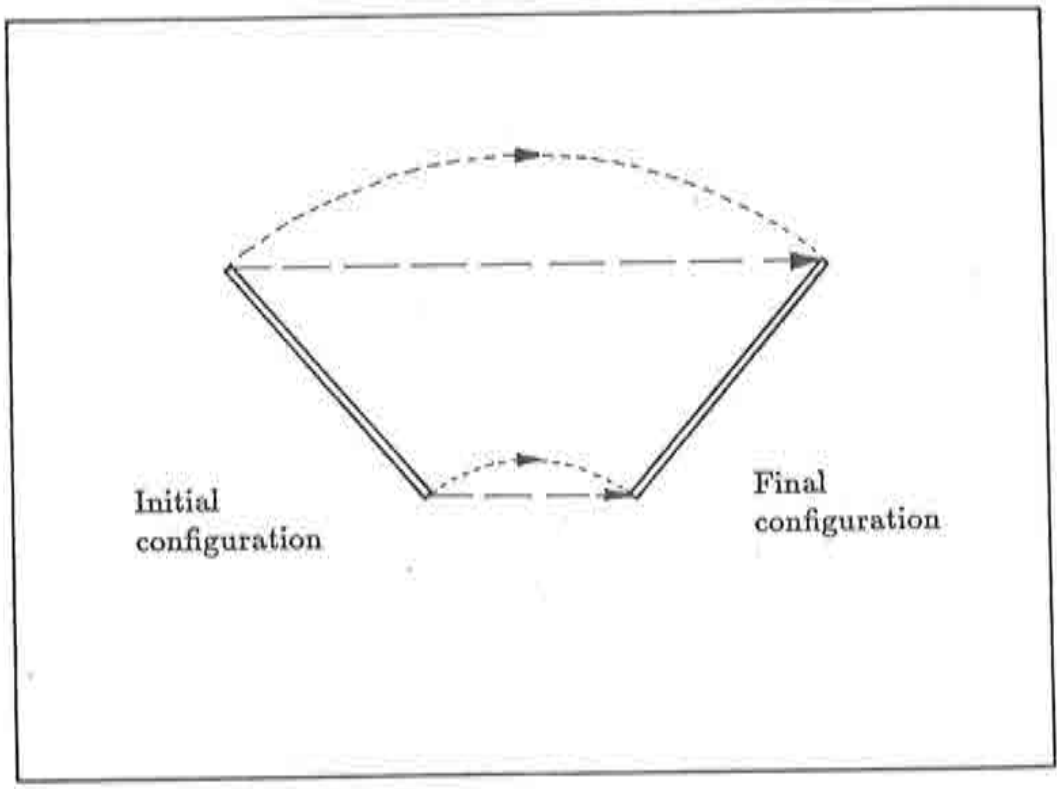


Figure 6: Two possible paths from an initial to a final configuration.

this analogy. Indeed, the incremental deformation gradient ${}^n\Lambda$ being an orthogonal tensor ${}^nR$ does not necessarily imply that the motion between t_n and t_{n+1} be a rigid rotation. Infinitely many other paths are possible, including such simple ones as the parallel translation of all the particles, depicted in Figure 6, which may cause loading/unloading effects. Since time is a discontinuous variable in the incremental analysis, nothing is known about the body motion during the time step. As a consequence, *assuming* a rigid rotation and demanding the numerical algorithm to perform in accordance to that assumption is an observer's choice. Although this choice is very reasonable, it has, contrary to the principle of objectivity, no physical justification. For this reason, the term *priority on rotation motion* (i.e., rigid rotations are assumed when possible) is preferred by the authors.

3.3 Two stress update algorithms in CASTEM2000

Current time integration algorithm (CEA) in CASTEM2000

It is possible to employ Eq. (40) as the strain measure in the increment Δt. The stress increment is then:

$$ {}^n\Delta\sigma = C : {}^nE \tag{42}$$

where the superscript n in $\Delta\sigma$ indicates that this tensorial quantity is, like ${}^nE$, referred to configuration Ω_n.

In a large-strain context it is no longer valid to compute the new stresses ${}^{n+1}\sigma$ by simply adding the stress increment ${}^n\Delta\sigma$ to the old stresses ${}^n\sigma$, because these latter two tensors are in the configuration Ω_n and ${}^{n+1}\sigma$ is sought in the configuration Ω_{n+1}. It is necessary to transform the tensors adequately by means of the push-forward Piola transformation, Eq. (17). The numerical algorithm for stress update is then:

$$ {}^{n+1}\sigma = {}^nJ^{-1}\ {}^n\Lambda \cdot {}^n\sigma \cdot {}^n\Lambda^T + {}^nJ^{-1}\ {}^n\Lambda \cdot ({}^n\Delta\sigma) \cdot {}^n\Lambda^T \tag{43}$$

where the Jacobian ${}^nJ$ is defined as $\det({}^n\Lambda)$ and the incremental deformation gradient, Eq. (39), is employed to push-forward both ${}^n\sigma$ and ${}^n\Delta\sigma$ into the new configuration Ω_{n+1}. From now on, the algorithm represented by Eq. (43) will be referred to as CEA.

The algorithm CEA treats rigid rotations properly: if ${}^n\Lambda$ is an orthogonal tensor, Eq. (40) yields a null strain tensor ${}^nE$ and Eq. (43) reduces to:

$$ {}^{n+1}\sigma = {}^nR \cdot {}^n\sigma \cdot {}^nR^T \tag{44}$$

thus predicting a rigid rotation of stresses, with no spurious stress variations. Note that the use of the full incremental Lagrange tensor, including second-order terms, is the key point of the incremental objectivity of this algorithm.

This algorithm, currently implemented in the object-oriented code CASTEM2000, can be programmed with a few lines of the GIBIANE meta-language, [4-7], provided by this code, as shown in Annex 1.

Although it gives priority to rigid rotation, as shown in Section 4.1, the algorithm CEA is not fully satisfactory. The choice of the configuration Ω_n to compute the stress increment, Eq. (42), induces an Euler-type, forward time integration, which limits the accuracy of the algorithm, as illustrated with various test cases in Section 4.

A new time integration algorithm (BCN) in CASTEM2000

An alternative, more accurate numerical algorithm for the time integration of the constitutive equation has been implemented in CASTEM2000. Following [11], the hypoelastic constitutive equation relates, via the modulus tensor $\boldsymbol{C}$, the rate of deformation tensor $\boldsymbol{d}$ to the Truesdell objective stress rate, Eq. (29c):

$$\boldsymbol{\sigma}_T^* = \boldsymbol{C} : \boldsymbol{d} \tag{45}$$

An intermediate configuration will be used to evaluate $\boldsymbol{d}$. Let ${}^n\boldsymbol{\eta}$ and ${}^{n+1}\boldsymbol{\eta}$ be the mappings of the material domain R_X into the configurations Ω_n and Ω_{n+1} of the spatial domain R_x, Eq. (1a). By linear interpolation, a one-parameter family of intermediate mappings ${}^{n+\alpha}\boldsymbol{\eta}$ and corresponding intermediate configurations $\Omega_{n+\alpha}$ may be defined:

$${}^{n+\alpha}\boldsymbol{\eta} = \alpha \ {}^{n+1}\boldsymbol{\eta} + (1-\alpha) \ {}^n\boldsymbol{\eta} \qquad 0 \leq \alpha \leq 1 \tag{46}$$

The associated deformation gradient is then

$${}^{n+\alpha}\boldsymbol{F} = \alpha \ {}^{n+1}\boldsymbol{F} + (1-\alpha) \ {}^n\boldsymbol{F} \tag{47}$$

and, similarly to Eq. (39), the incremental deformation gradient relating the intermediate and final configurations is:

$${}^{n+\alpha}\boldsymbol{\Lambda} = {}^{n+1}\boldsymbol{F} \,.\, {}^{n+\alpha}\boldsymbol{F}^{-1} \tag{48}$$

with the Jacobian ${}^{n+\alpha}J$ defined as $\det\left({}^{n+\alpha}\boldsymbol{\Lambda}\right)$. The different deformation gradient tensors are summarized in Figure 7.

By using a generalized midpoint rule algorithm, the stress update becomes

$${}^{n+1}\boldsymbol{\sigma} = {}^nJ^{-1} \ {}^n\boldsymbol{\Lambda} \,.\, {}^n\boldsymbol{\sigma} \,.\, {}^n\boldsymbol{\Lambda}^T + {}^{n+\alpha}J^{-1} \ {}^{n+\alpha}\boldsymbol{\Lambda} \cdot (\Delta t \ \boldsymbol{C} : \boldsymbol{d})\,|_{n+\alpha} \,.\, {}^{n+\alpha}\boldsymbol{\Lambda}^T \tag{49}$$

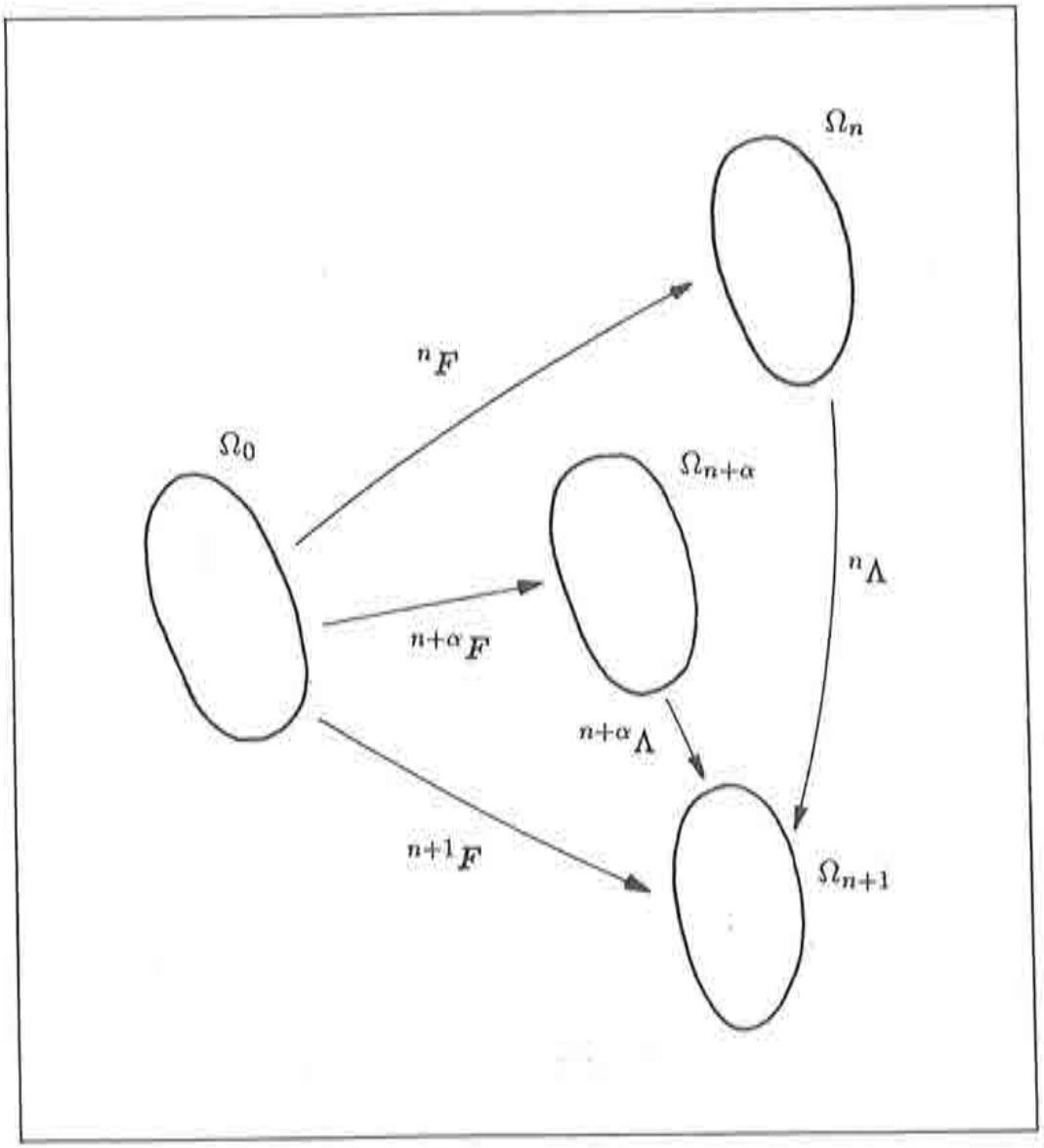


Figure 7: Deformation gradients in incremental analysis involving an intermediate configuration $\Omega_{n+\alpha}$.

As in Eq. (43), tensors referred to the initial and intermediate configurations are pushed forward into the final one by means of the appropriate incremental deformation gradients ${}^{n}\Lambda$ and ${}^{n+\alpha}\Lambda$. This new algorithm will be referred to as BCN.

Recalling the definition of $\boldsymbol{d}$, Eq. (8), the approximation to ${}^{n+\alpha}\boldsymbol{d}$ needed in Eq. (49) will be:

$$ {}^{n+\alpha}d = \frac{1}{\Delta t}\left[\frac{\partial \Delta \boldsymbol{u}}{\partial^{n+\alpha}\boldsymbol{x}}\right]^{S} \tag{50} $$

where $\Delta \boldsymbol{u} = {}^{n+1}\boldsymbol{\eta} - {}^{n}\boldsymbol{\eta}$ is the incremental material displacement and superscript S means "symmetric part".

The stress increment is then

$$ {}^{n+\alpha}\Delta\boldsymbol{\sigma} = (\Delta t\ \boldsymbol{C} : \boldsymbol{d})\,|_{n+\alpha} = \boldsymbol{C}\left[\frac{\partial \Delta \boldsymbol{u}}{\partial\ {}^{n+\alpha}\boldsymbol{x}}\right]^{S} \tag{51} $$

and can be computed as in a small strain analysis, since only first-order terms are present in Eq. (50).

This new stress update algorithm BCN can also be implemented in a straightforward manner in CASTEM2000, as shown in Annex 2.

The algorithm BCN treats rotations correctly if and only if the midstep configuration ($\alpha = 0.5$) is chosen as the intermediate one, as proven in [11]. In the rigid rotation test of Section 4.1, good results are obtained for $\alpha = 0.5$. For different values of α, spurious stress increments are predicted.

4. COMPARISON OF TWO TIME-INTEGRATION ALGORITHMS

4.0 Introduction

The two algorithms presented in Section 3.3, CEA and BCN, are compared with the help of various simple deformation paths: rigid rotation, simple shear, uniaxial extension, extension and compression, dilatation. Both elastic and elastoplastic cases are considered.

Elastic behaviour will be represented by Young's modulus E and Poisson's coefficient ν or, equivalently, by the Lamé constants λ and μ:

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)} \tag{52}$$

$$\mu = \frac{E}{2(1+\nu)}$$

All the tests have been performed with $\nu = 0$, resulting in $\lambda = 0$ and $\mu = E/2$.

The rigid rotation test demonstrates that, as predicted by theory, both algorithms treat rigid rotations properly. It is also shown that BCN performs well if and only if the midstep configuration ($\alpha = 0.5$) is chosen as the intermediate configuration in Eq. (46).

Differences between CEA and BCN arise for the other, non-rigid, test cases. In the simple shear test, for instance, CEA predicts unrealistic vertical normal stresses, while BCN yields correct null values. The results depend heavily on the number of time increments if CEA is employed, and only slightly with BCN. This behaviour is confirmed by the other deformation paths.

4.1 Rigid rotation

A rectangle $ABCD$ is initially subjected to a uniaxial stress field:

$$\sigma_{xx} = 1. \qquad (53)$$

$$\sigma_{xy} = 0.$$

$$\sigma_{yy} = 0.$$

and rotated 90° around vertex A. Both CEA and BCN are capable of rotating the stress field accordingly:

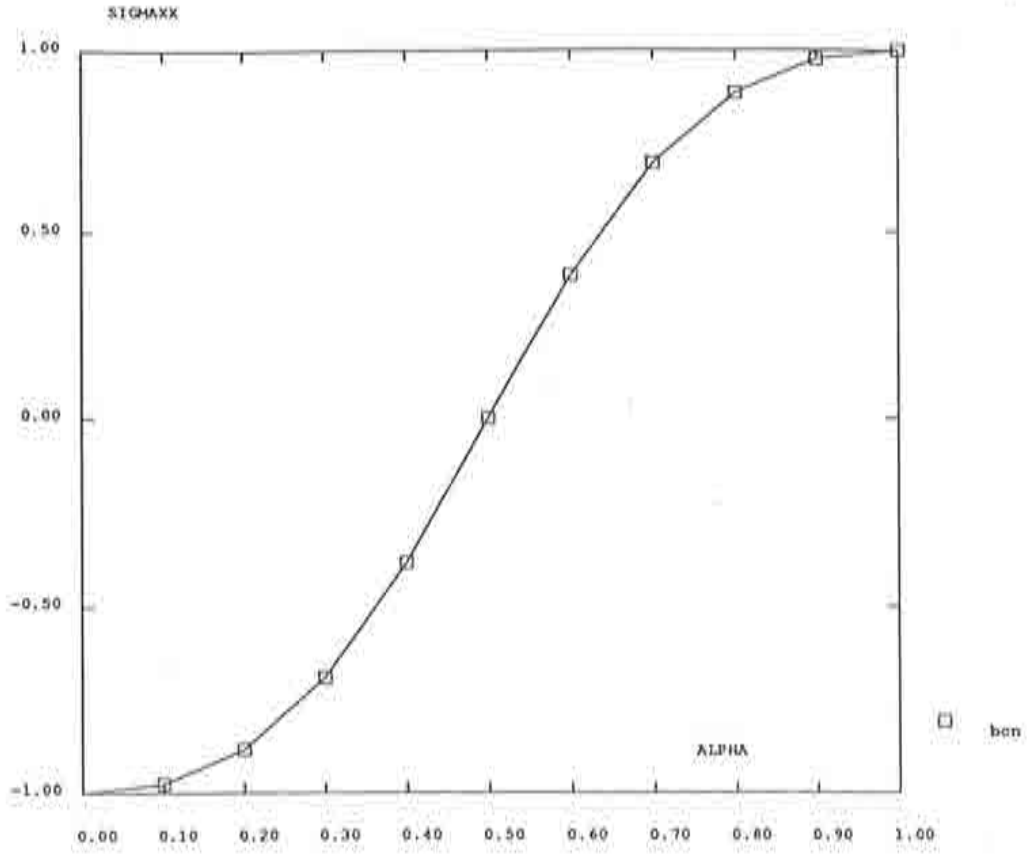
$$\sigma_{xx} = 0. \qquad (54)$$

$$\sigma_{xy} = 0.$$

$$\sigma_{yy} = 1.$$

It has been commented above and proved in [11] that the choice of the midstep configuration ($\alpha = 0.5$) as the intermediate one is essential to the objectivity of BCN. This fact can be corroborated by a simple test, where various intermediate configurations, ranging from the initial one ($\alpha = 0.$) to the final one ($\alpha = 1.$), are employed. Figures 8a and 8b show the horizontal (σ_{xx}) and vertical (σ_{yy}) normal stress respectively versus α. It can be seen that only the choice $\alpha = 0.5$ yields the correct results of Eq. (54). For different values of α, spurious stress variations appear.

a)



b)

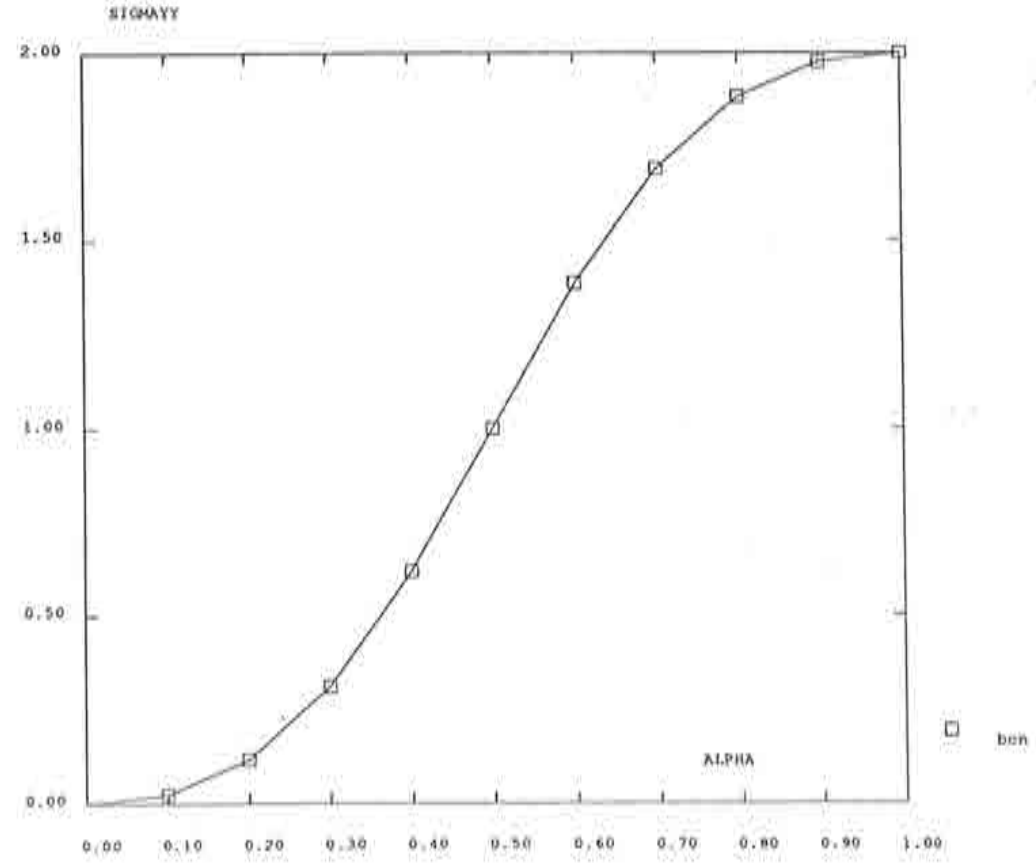


Figure 8: Rigid rotation test. Influence of intermediate configuration. a) σ_{xx} vs. α. b) σ_{yy} vs. α.

4.2 Simple shear

The problem statement for the simple shear test is shown in Figure 9, where the initial ($t = 0$) and final ($t = 1$) configurations are presented. The initial stress field is zero. The equations of motion are:

$$x(t) = X + Yt \tag{55}$$

$$y(t) = Y$$

and the deformation gradient is:

$$F = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \tag{56}$$

If the Truesdell objective rate is employed, Eq. (45) yields the system of ordinary differential equations:

$$\dot{\sigma}_{xx} - 2\sigma_{xy} = 0 \tag{57}$$

$$\dot{\sigma}_{xy} - \sigma_{yy} = E/2$$

$$\dot{\sigma}_{yy} = 0$$

Complemented with null initial conditions, Eq. (57) may be solved to provide the analytical solution:

$$\sigma_{xx}(t) = \frac{E}{2}t^2 \tag{58}$$

$$\sigma_{xy}(t) = \frac{E}{2}t$$

$$\sigma_{yy}(t) = 0$$

Both algorithms CEA and BCN have been employed, with different values of the time step (1, 2, 3, 5, 10, 20, 50 increments), to integrate the constitutive equation for the given deformation path, Eq. (55), from $t = 0$ to $t = 1$.

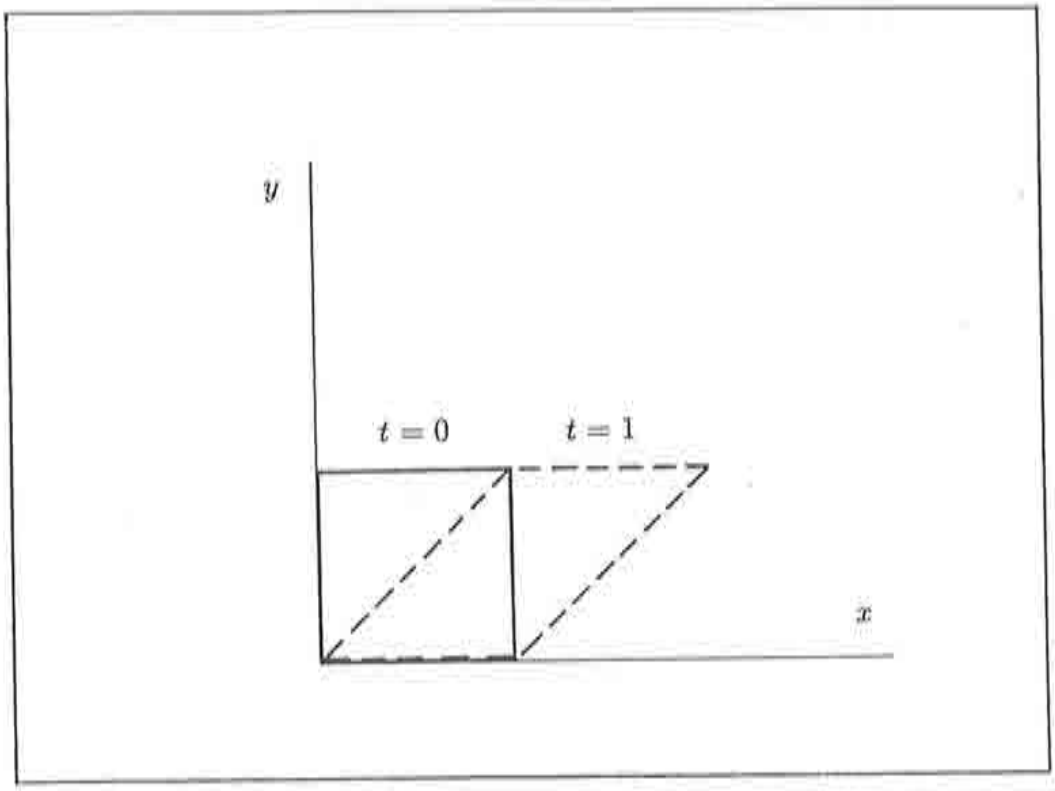


Figure 9: Simple shear test. Initial and final configurations.

Figure 10 presents the results for the elastic case (1, 5, 50 increments). It can be seen that BCN gets, for the three components of stress, the exact analytical values, whereas CEA grossly overestimates stresses when not enough increments are employed, and demands a small time-step to get close to the analytical solution. The output of a small strain analysis (1 increment), is also presented in Figure 10: the null σ_{yy} and linear σ_{xy} are correctly predicted, but this linear analysis is not able to reproduce quadratic σ_{xx}, Eq. (58), caused by a second-order strain.

Figure 11 presents an alternative view, where the final stress ($t = 1$) is depicted versus the number of time increments. With 50 time increments, CEA still supplies a solution which is slightly different from the exact one, captured by BCN with a single time-step. This fact is especially relevant concerning the vertical stress σ_{yy}: CEA leads to unrealistic non-zero values, that decrease but are not completely cancelled when the time-step is reduced, see Figures 10c and 11c.

A similar analysis has been carried out in the elastoplastic case. The constitutive law is assumed to be bilinear, see Figure 12, with

$$E_p = E/100 \tag{59}$$

$$\sigma_y = E/2$$

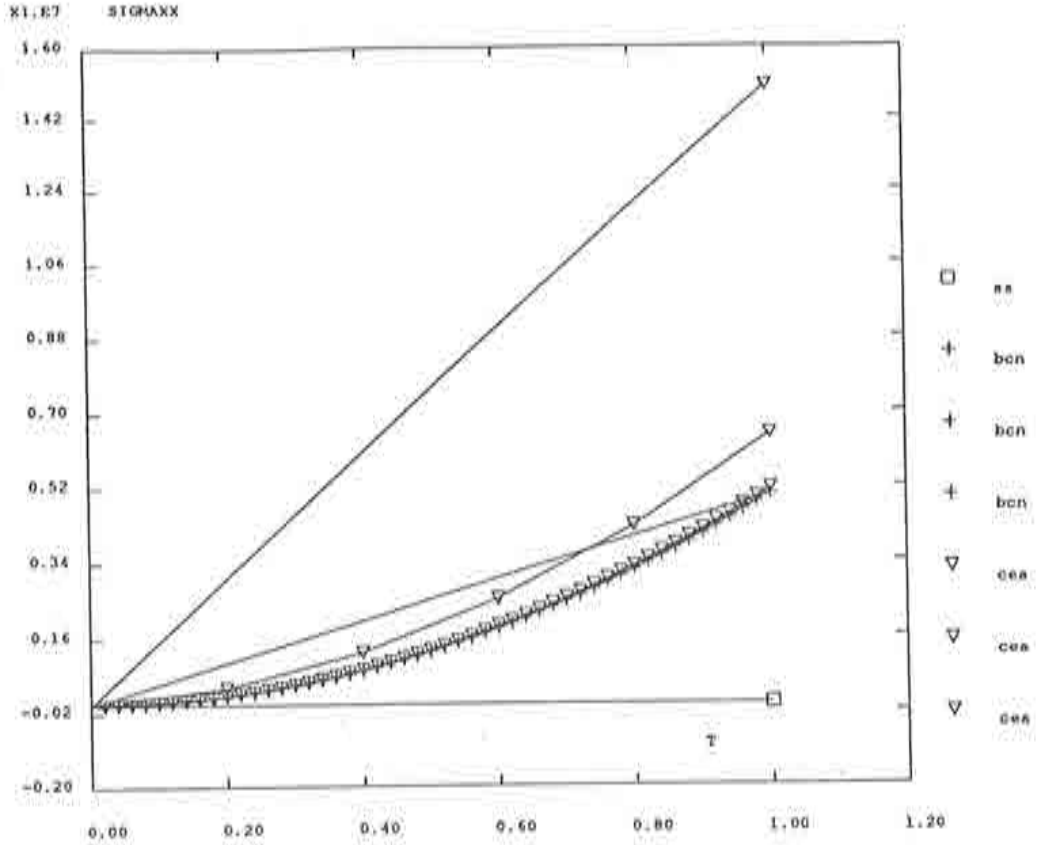
where E_p is the plastic modulus and σ_y is the yield stress. The results are presented in Figure 13 (1, 10, 50 increments). As in the elastic problem, CEA grossly overestimates the final stress if only one increment is employed, while BCN predicts much more accurate values. With a higher number of time-steps, both algorithms converge to the same response. It may be observed that, of course, when plastification starts (around $t = 0.6$), the curves differ from their elastic counterparts of Figure 10.

A small strain analysis, this time with 50 increments to account for material nonlinearity –plasticity–, is again not satisfactory. As commented above, σ_{xx} is uncorrectly kept at its zero initial value throughout the first steps of the computation, with elastic behaviour. As σ_{yy} is also zero, the only non-zero stress is σ_{xy}, and the elastic stress tensor is

$$\boldsymbol{\sigma} = \begin{bmatrix} 0 & \sigma_{xy} \\ \sigma_{xy} & 0 \end{bmatrix} \tag{60}$$

that is, a fully deviatoric tensor. As the plastic correction, once plastification begins, is carried along precisely along the deviatoric part of the stress tensor, σ_{xy} is the only component of stress affected by plastification, while σ_{xx} and σ_{yy} keep their elastic behaviour. The need for a large strain analysis is again demonstrated.

a)



b)

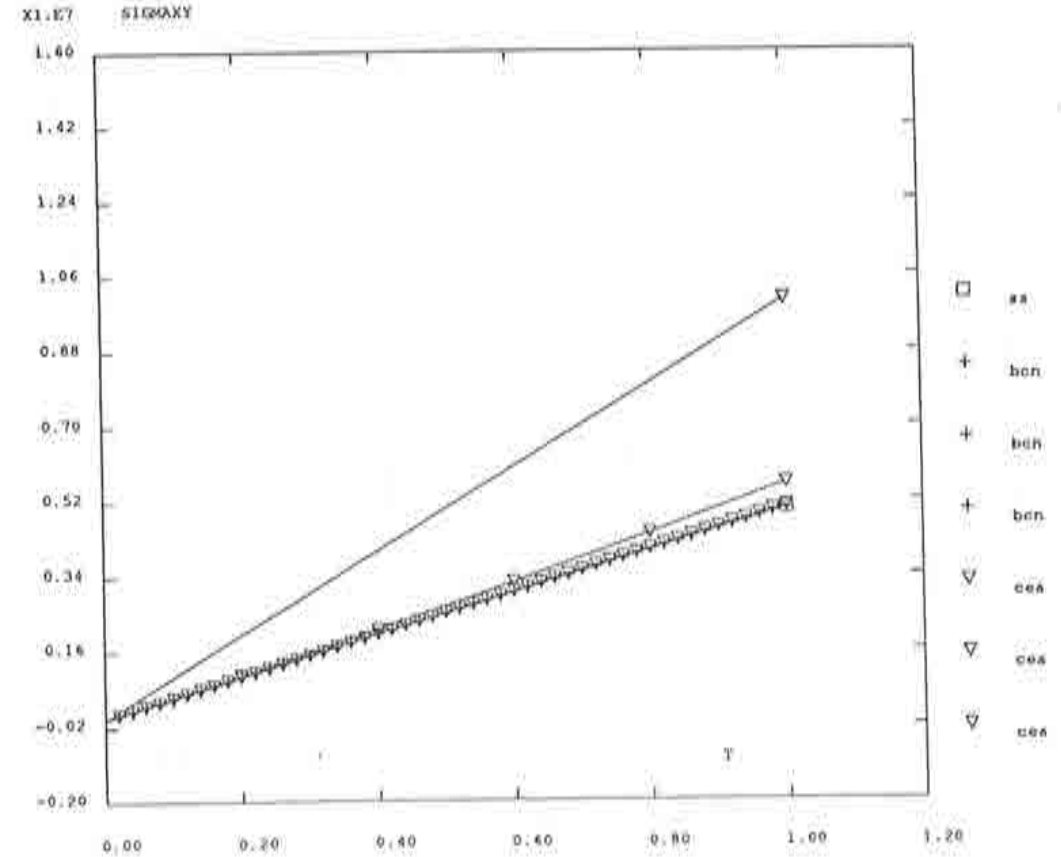


Figure 10: Simple shear test, elastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 5 and 50 time steps) and SS (small strain, 1 time step). a) σ_{xx}. b) σ_{xy}.

c)

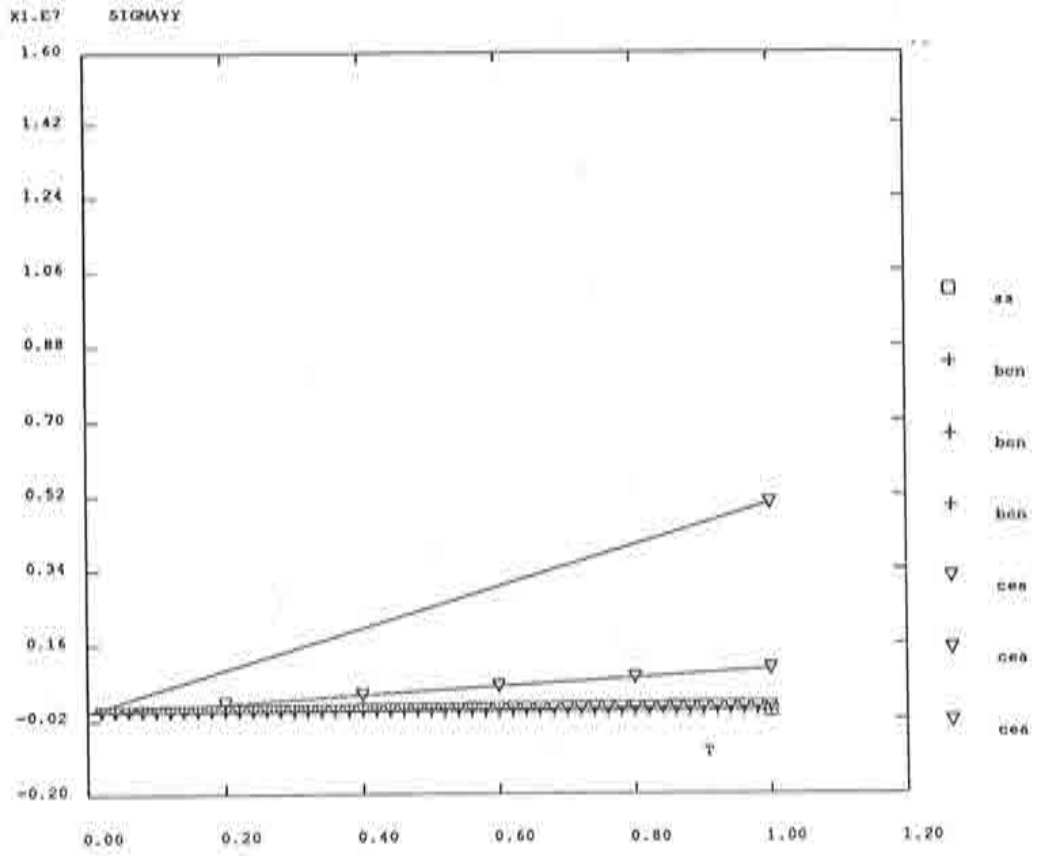
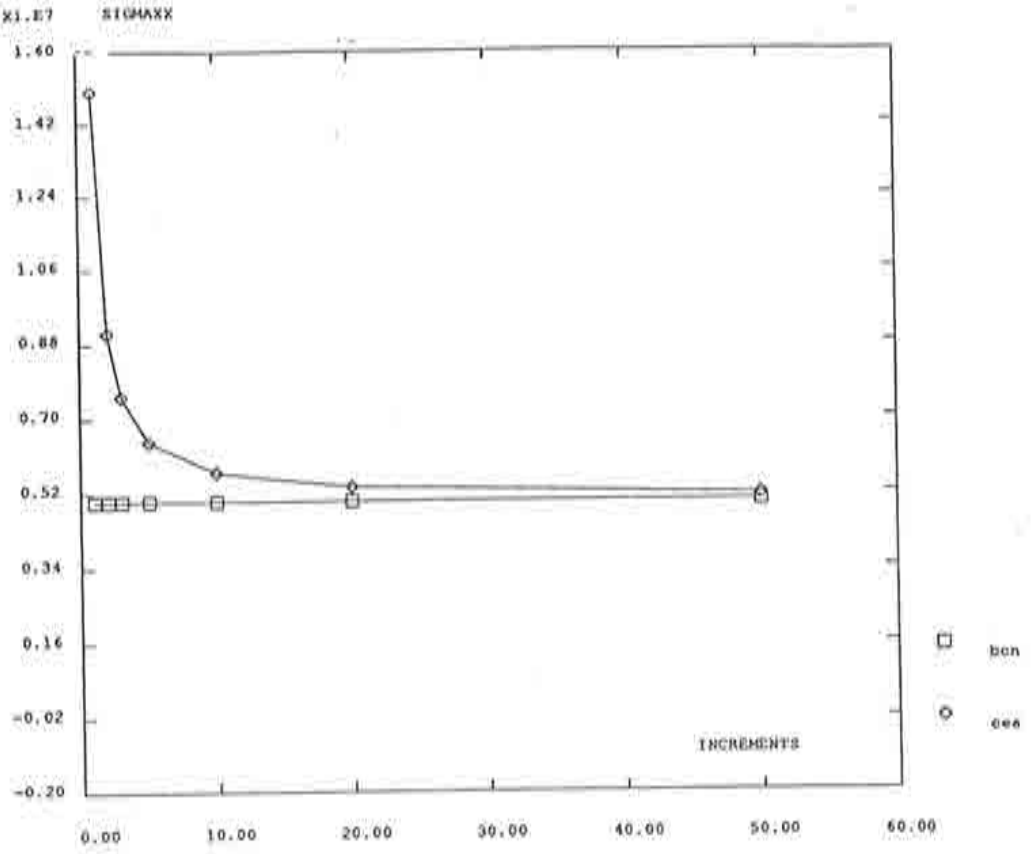


Figure 10 (continued): Simple shear test, elastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 5 and 50 time steps) and SS (small strain, 1 time step). c) σ_{yy}.

a)



b)

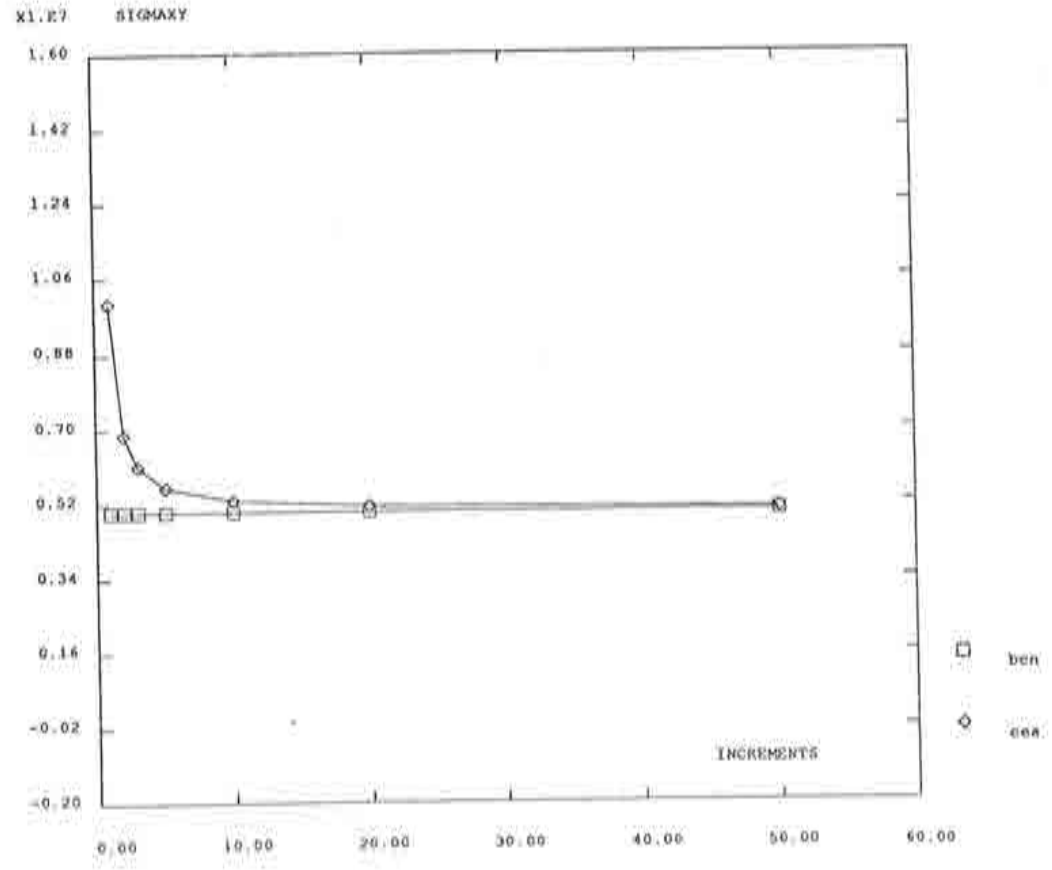


Figure 11: Simple shear test, elastic analysis. Final value of stress ($t = 1$) vs. number of time steps (1, 2, 3, 5, 10, 20, 50). a) σ_{xx}. b) σ_{xy}.

c)

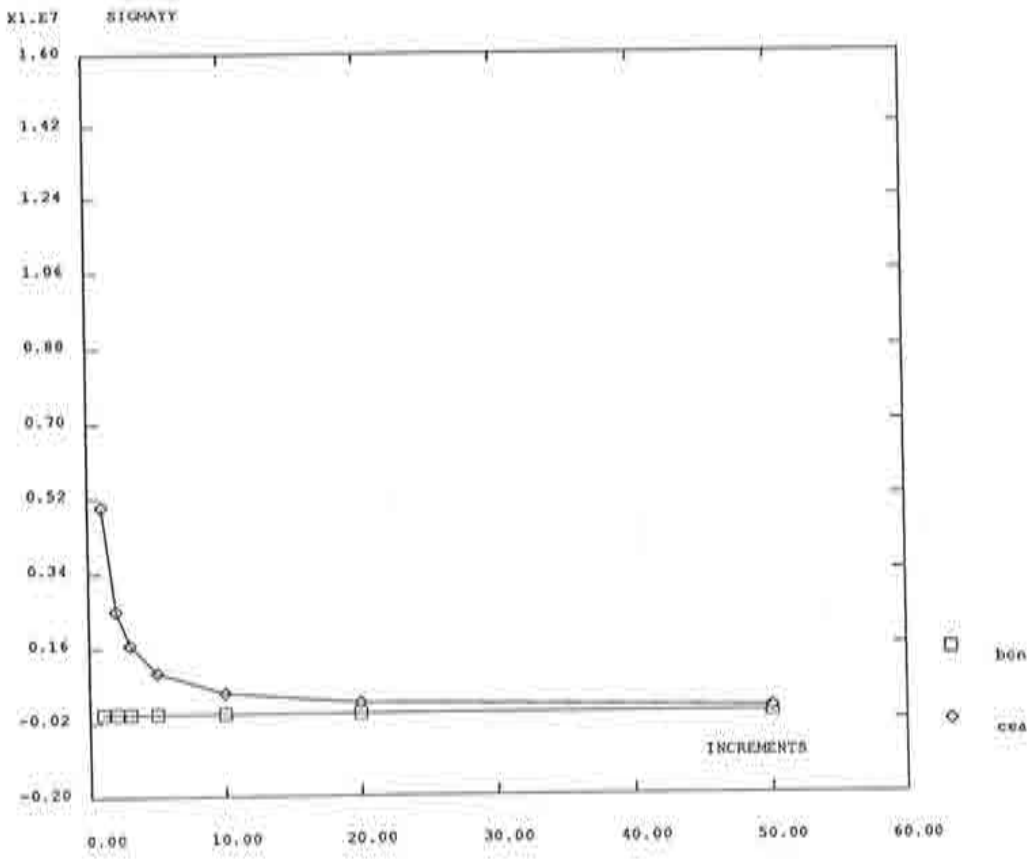


Figure 11 (continued): Simple shear test, elastic analysis. Final value of stress ($t = 1$) vs. number of time steps (1, 2, 3, 5, 10, 20, 50). c) σ_{yy}.

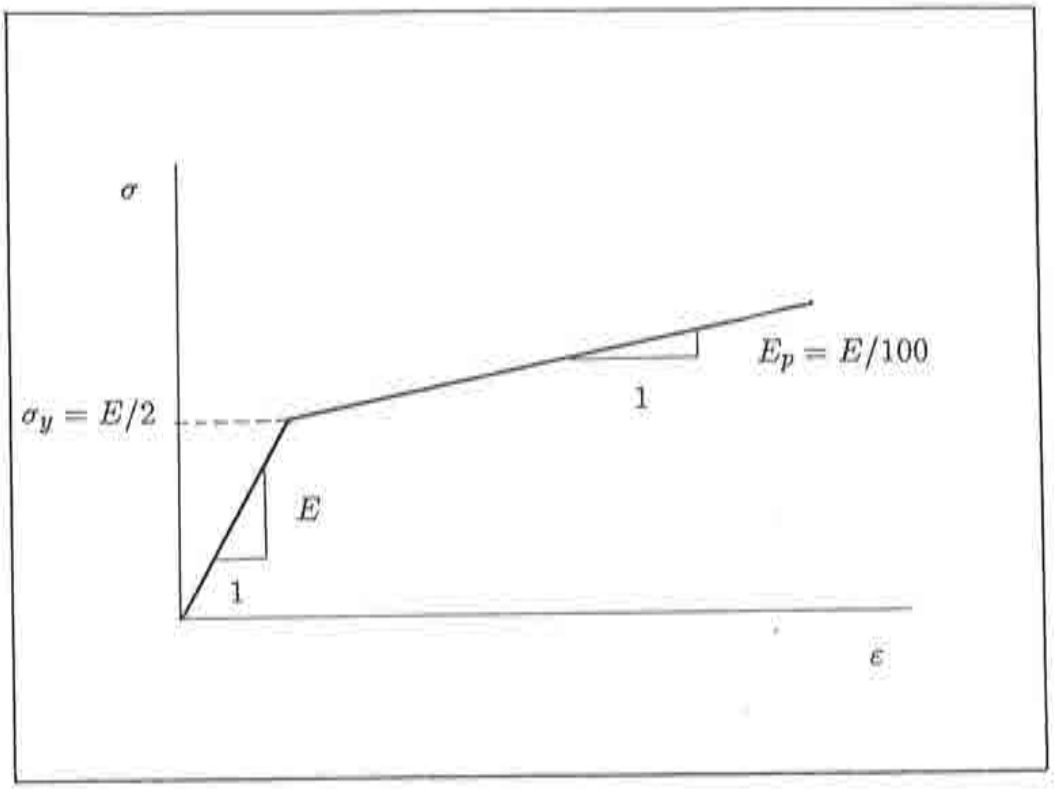
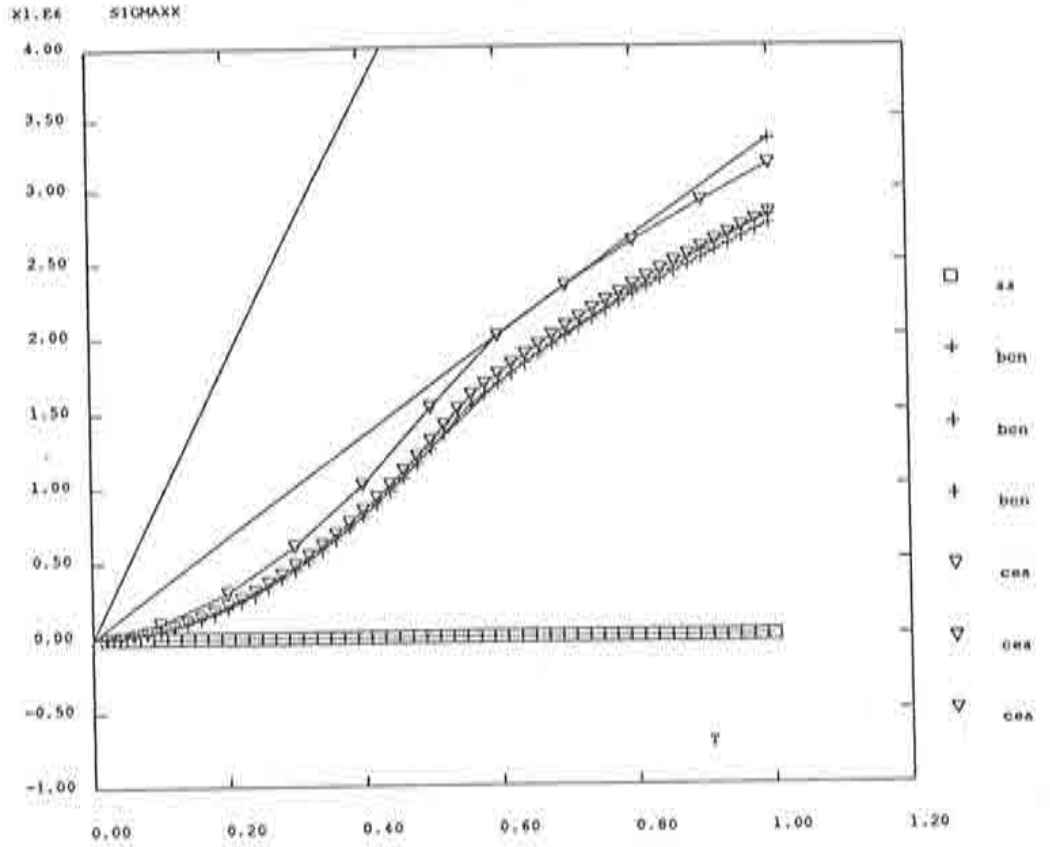


Figure 12: Bilinear elastoplastic constitutive equation.

a)



b)

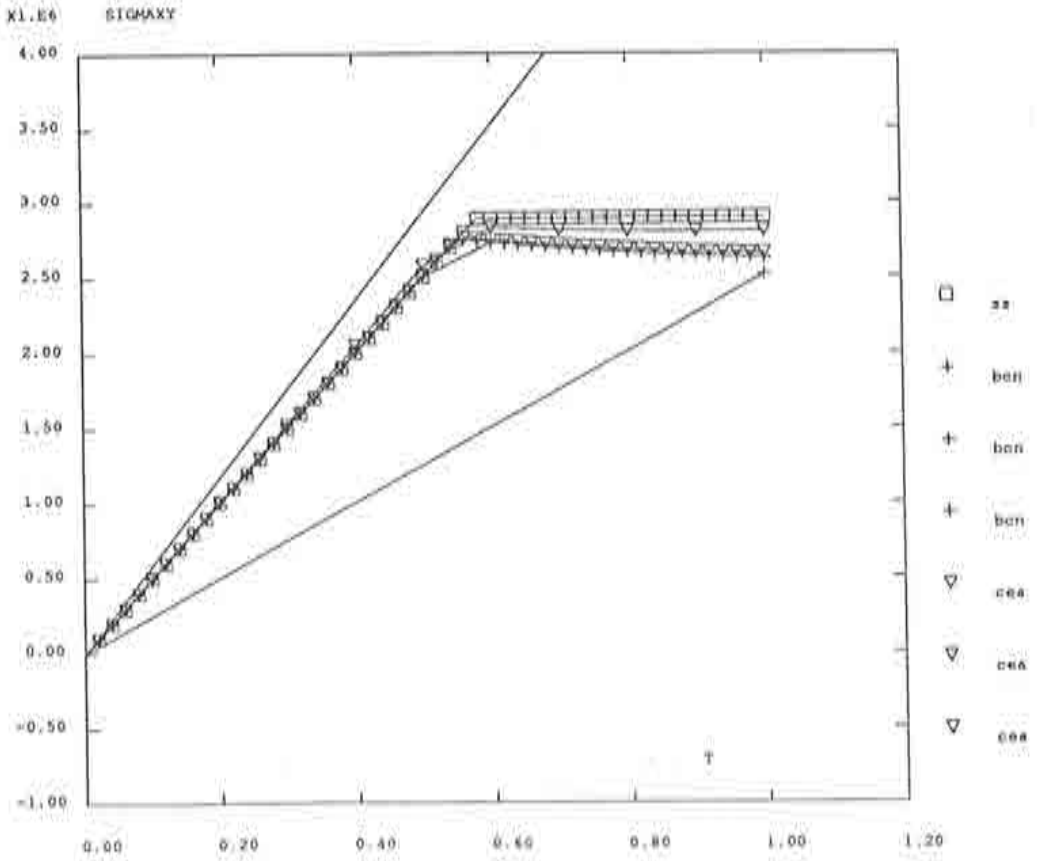


Figure 13: Simple shear test, elastoplastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 10 and 50 time steps) and SS (small strain, 50 time steps). a) σ_{xx}. b) σ_{xy}.

c)

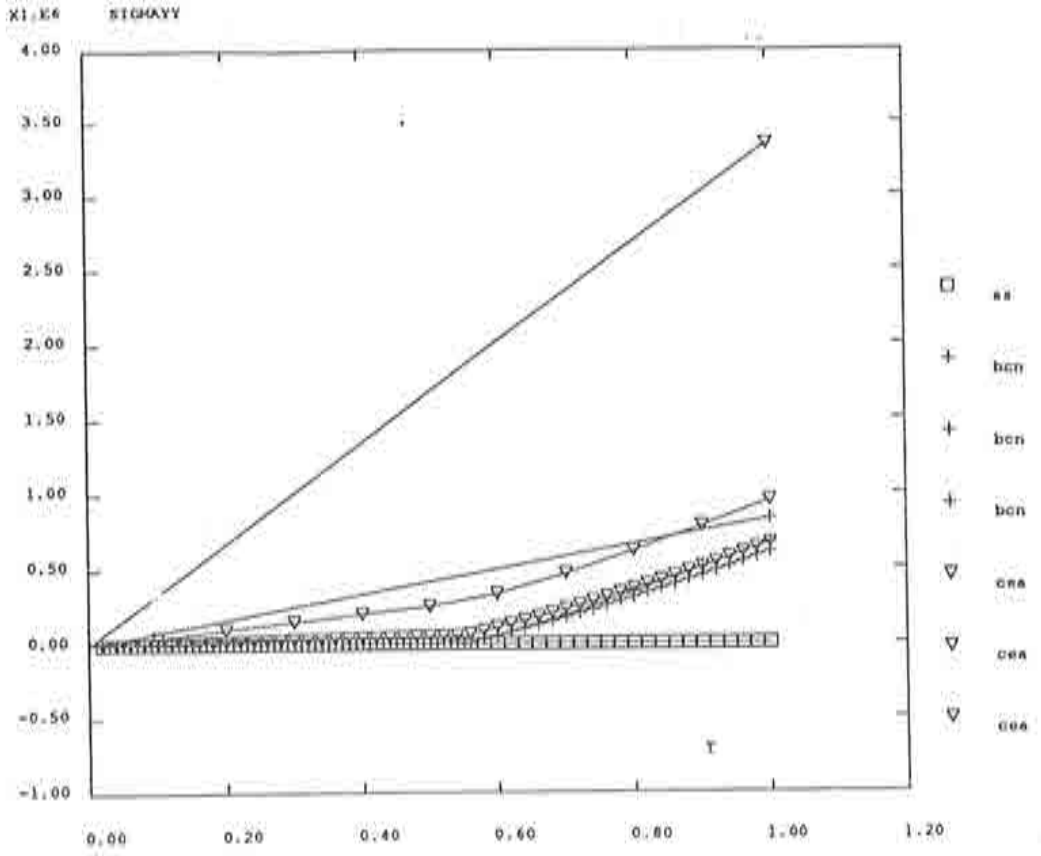


Figure 13 (continued): Simple shear test, elastoplastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 10 and 50 time steps) and SS (small strain, 50 time steps). c) σ_{yy}.

4.3 Uniaxial extension

A unit square is subjected to uniaxial extension in the x direction, see Figure 14. The equations of motion are:

$$x(t) = X\,(1+t) \tag{61}$$

$$y(t) = Y$$

and the deformation gradient is:

$$F = \begin{bmatrix} 1+t & 0 \\ 0 & 1 \end{bmatrix} \tag{62}$$

The analytical solution of Eq. (45) is, this case:

$$\sigma_{xx}(t) = Et \tag{63}$$

$$\sigma_{xy}(t) = 0$$

$$\sigma_{yy}(t) = 0$$

Both CEA and BCN correctly predict null values for σ_{xy} and σ_{yy}. Differences are found, on the contrary, for σ_{xx}: while CEA grossly overestimates it, BCN slighlty underestimates it, see Figures 15 and 16. It can be seen in Figure 15 that, for this particular test, a small strain analysis, with one time-step, produces the exact solution. This is due to the fact that the two sources of error (neglecting second-order terms in the strain tensor and the time-discretization error) compensate each other. Of course, this is not the situation in a general case, as shown for the other tests.

The plastic response is presented in Figure 17. When plastification begins at $t = 0.5$, the plastic correction is performed along the deviatoric part of the stress tensor, thus resulting in an increase of σ_{yy} and the expense of σ_{xx}. Again, BCN behaves much better than CEA if a small number of time-increments is employed, especially for σ_{xx}, see Figure 17a. With a large number of time-steps (50), CEA and BCN provide very similar results, different from the small strain analysis, see Figure 17b.

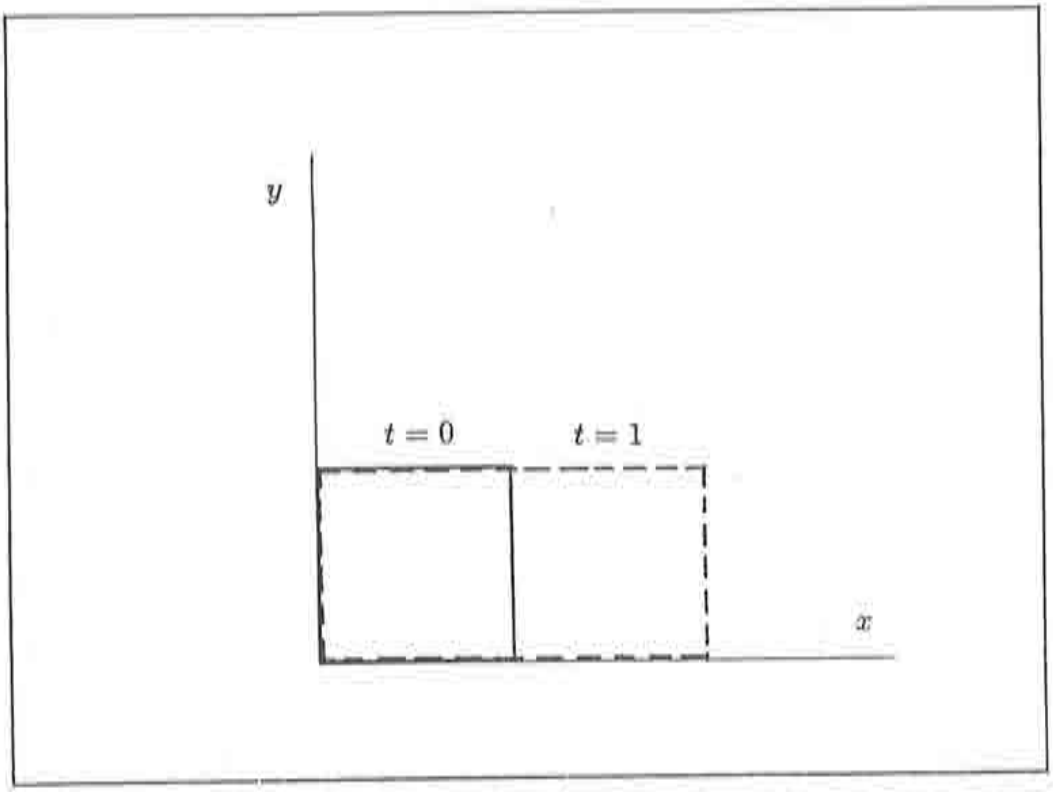


Figure 14: Uniaxial extension test. Initial and final configurations.

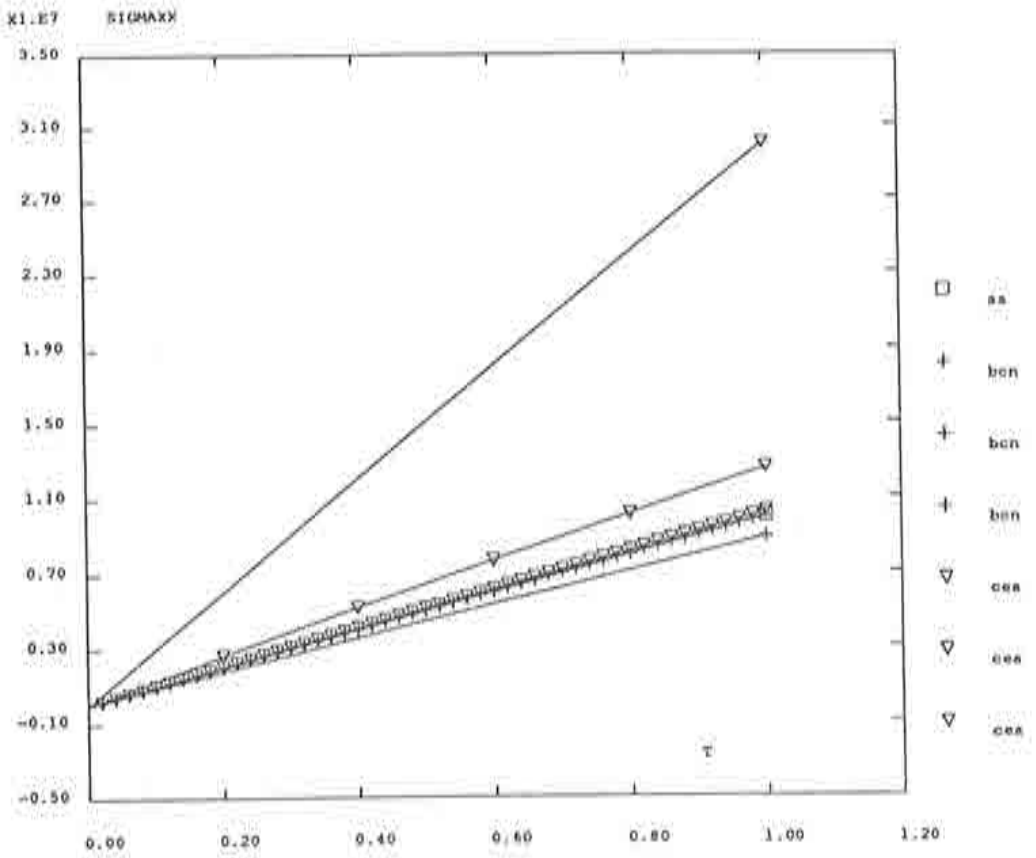


Figure 15: Uniaxial extension test, elastic analysis. Horizontal normal stress σ_{xx} vs. time curves computed with algorithms CEA and BCN (1, 5 and 50 time steps) and SS (small strain, 1 time step).

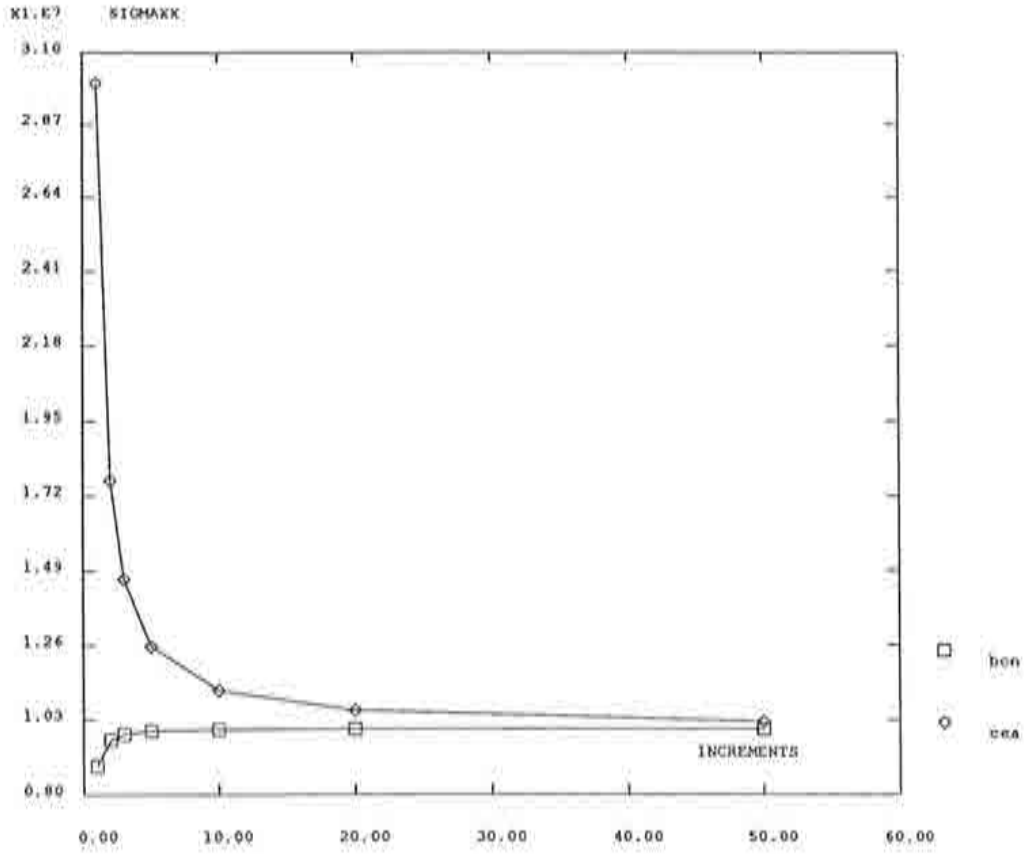
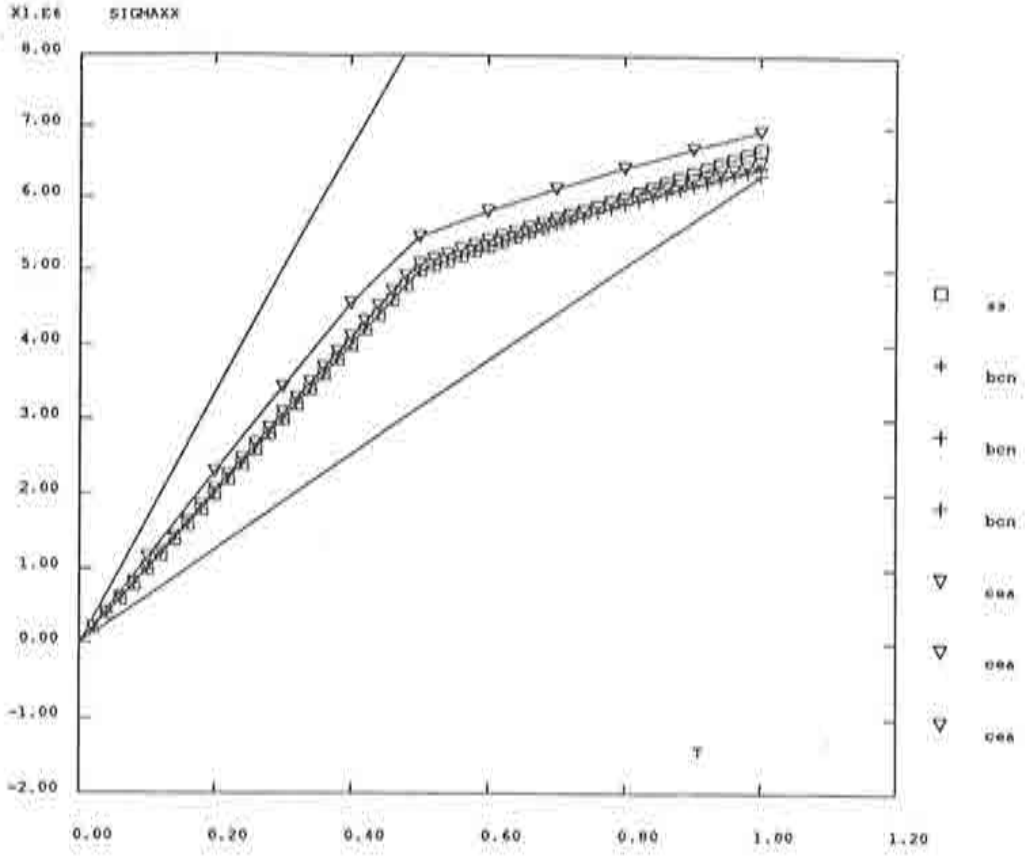


Figure 16: Uniaxial extension test, elastic analysis. Final value of horizontal normal stress σ_{xx} ($t = 1$) vs. number of time steps (1, 2, 3, 5, 10, 20, 50).

a)



b)

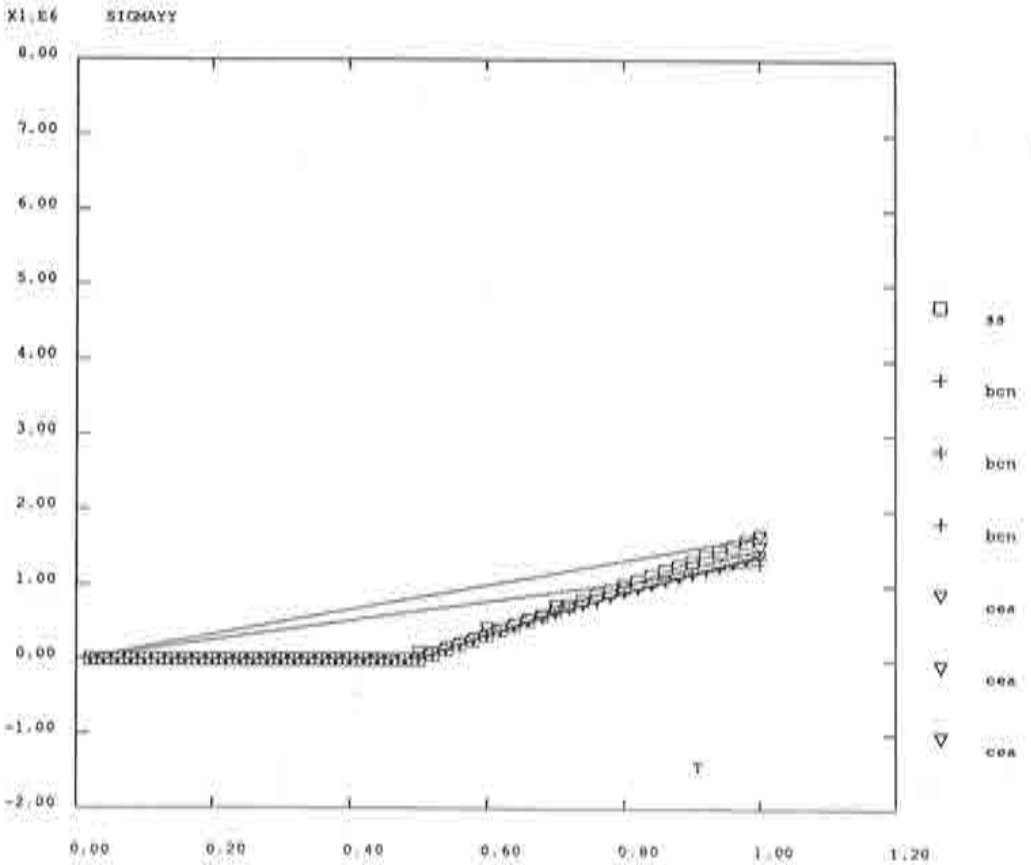


Figure 17: Uniaxial extension test, elastoplastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 10 and 50 time steps) and SS (small strain, 50 time steps). a) σ_{xx}. b) σ_{yy}.

4.4 Extension and compression

A unit square undergoes extension in the x direction and compression in the y direction, with no change in volume, see Figure 18. The equations of motion are:

$$x(t) = X\,(1+t) \tag{64}$$

$$y(t) = Y/\,(1+t)$$

and the deformation gradient is:

$$F = \begin{bmatrix} 1+t & 0 \\ 0 & 1/1+t \end{bmatrix} \tag{65}$$

and the analytical solution of Eq. (45) is:

$$\sigma_{xx}(t) = E\left(t + \frac{t^2}{2}\right) \tag{66}$$

$$\sigma_{xy}(t) = 0$$

$$\sigma_{yy}(t) = \frac{E}{2}\left[\frac{1}{(1+t)^2 - 1}\right]$$

Again, both CEA and BCN are capable of predicting null σ_{xy} for the elastic test, but differences appear for σ_{xx} and σ_{yy}, see Figures 19 and 20.

As for the plastic test, results are shown in Figure 21. Once again, convergence to the "reference" solution (with 50 time increments) is faster with BCN than with CEA, while the small strain analysis yields a qualitatively different response.

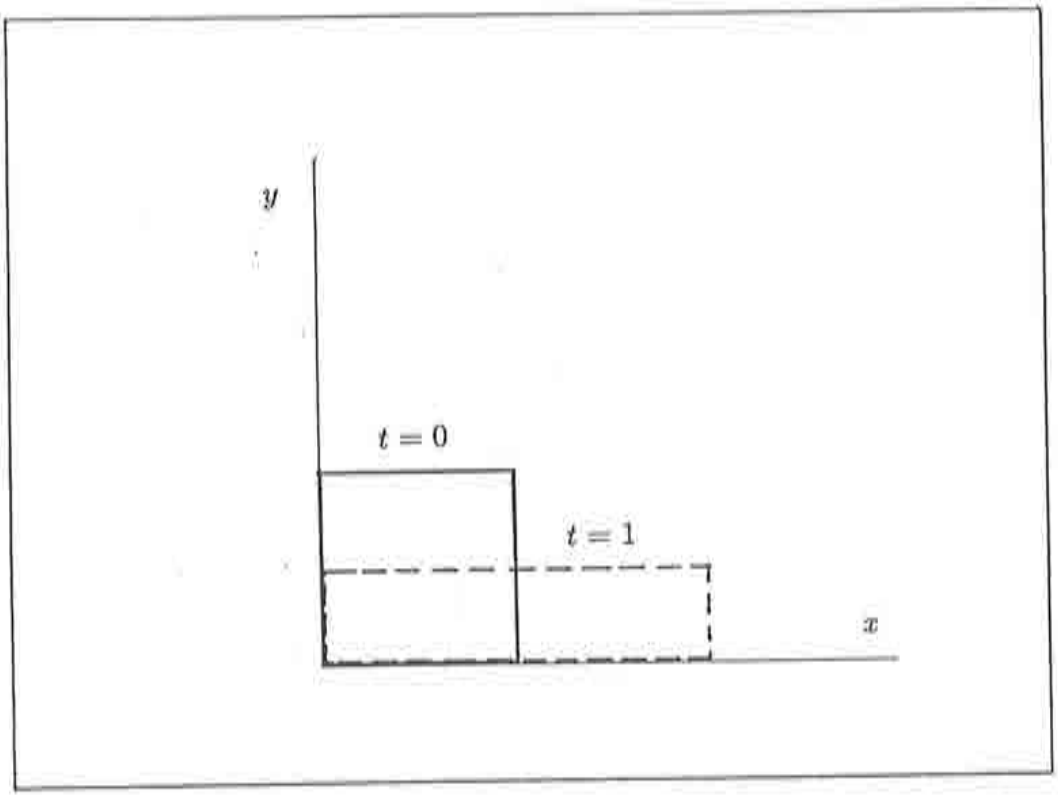
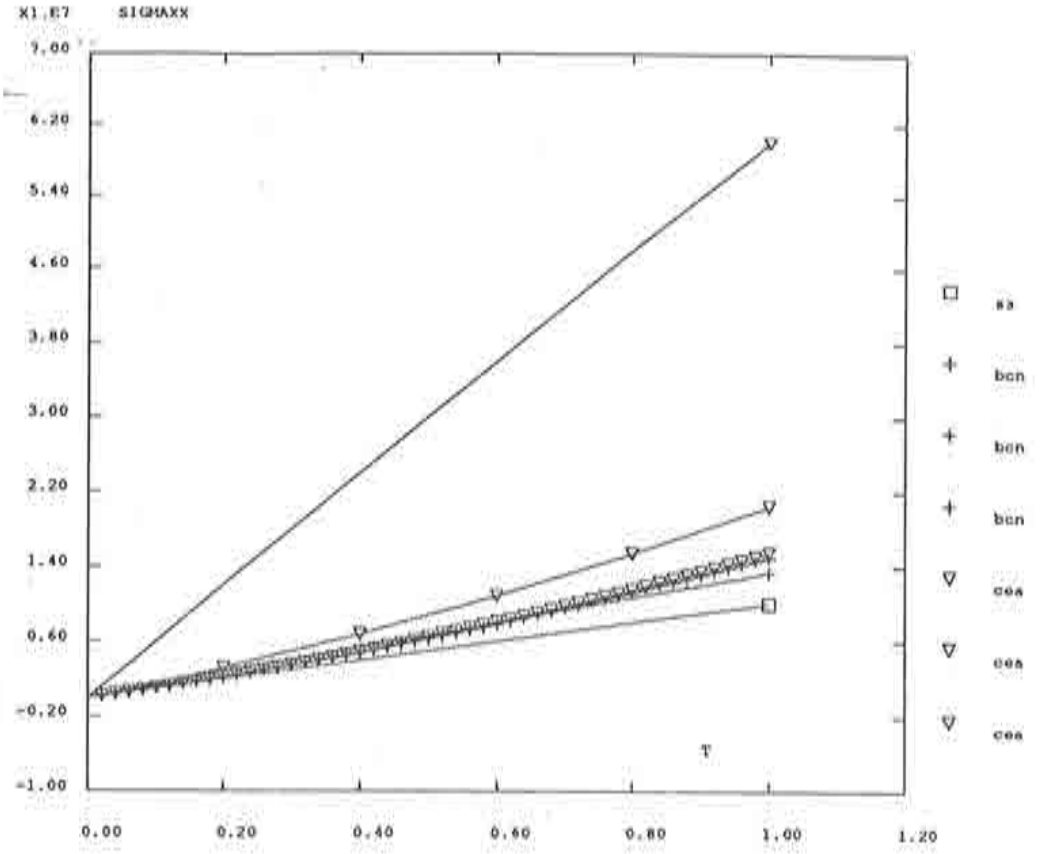


Figure 18: Extension and compression test. Initial and final configurations.

a)



b)

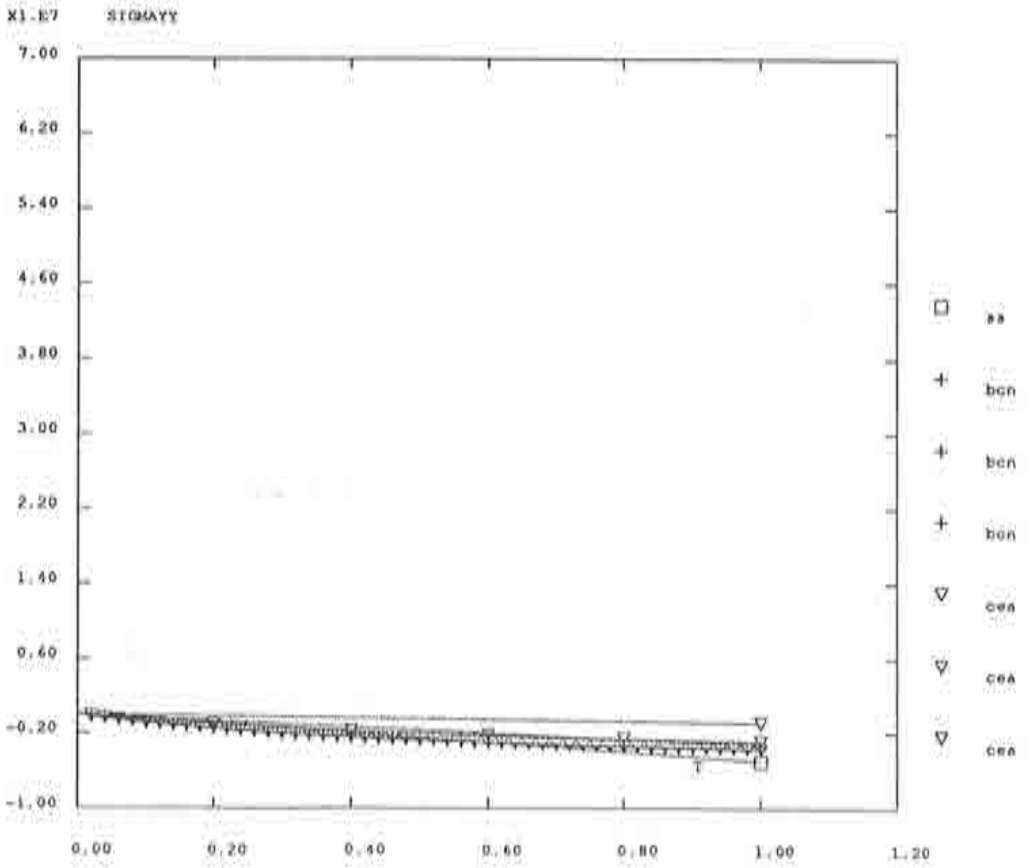
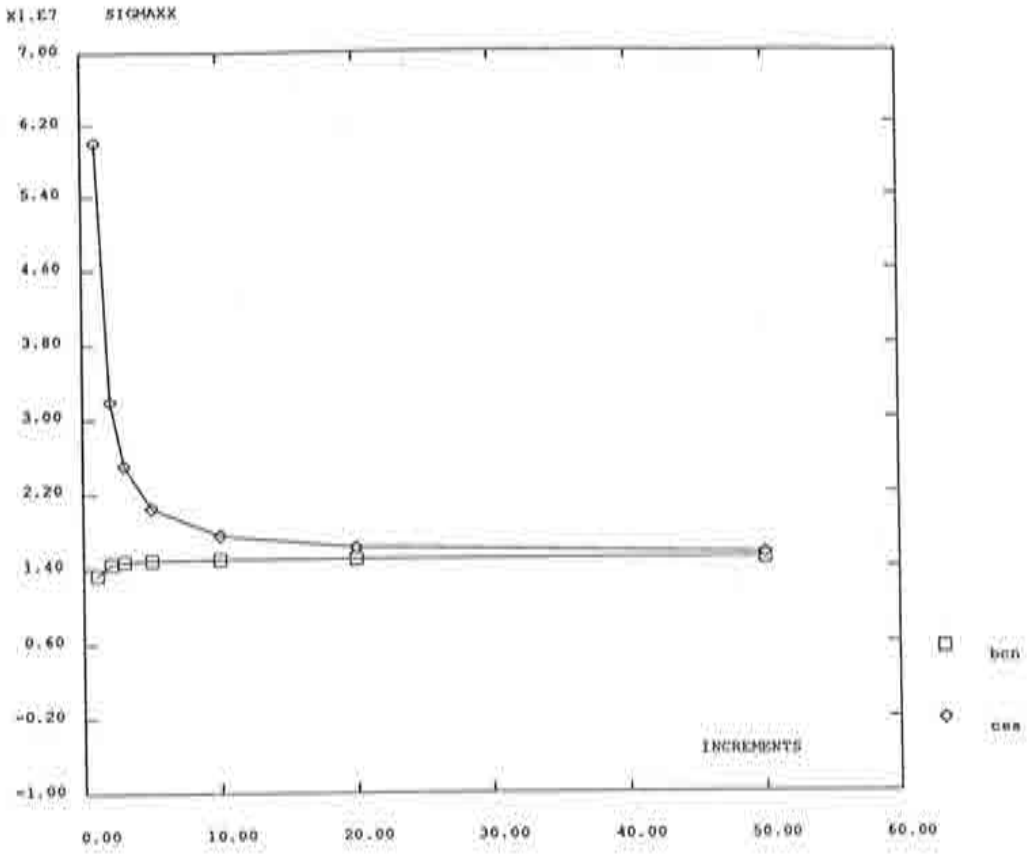


Figure 19: Extension and compression test, elastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 5 and 50 time steps) and SS (small strain, 1 time step). a) σ_{xx}. b) σ_{yy}.

a)



b)

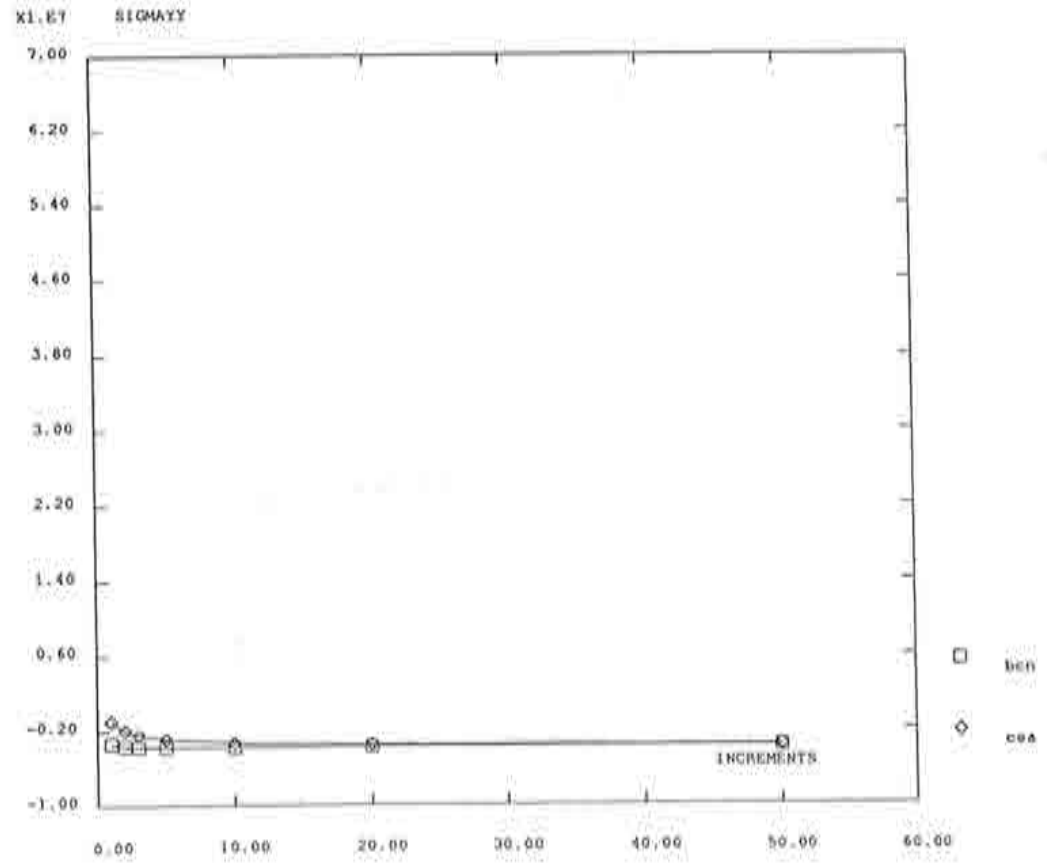
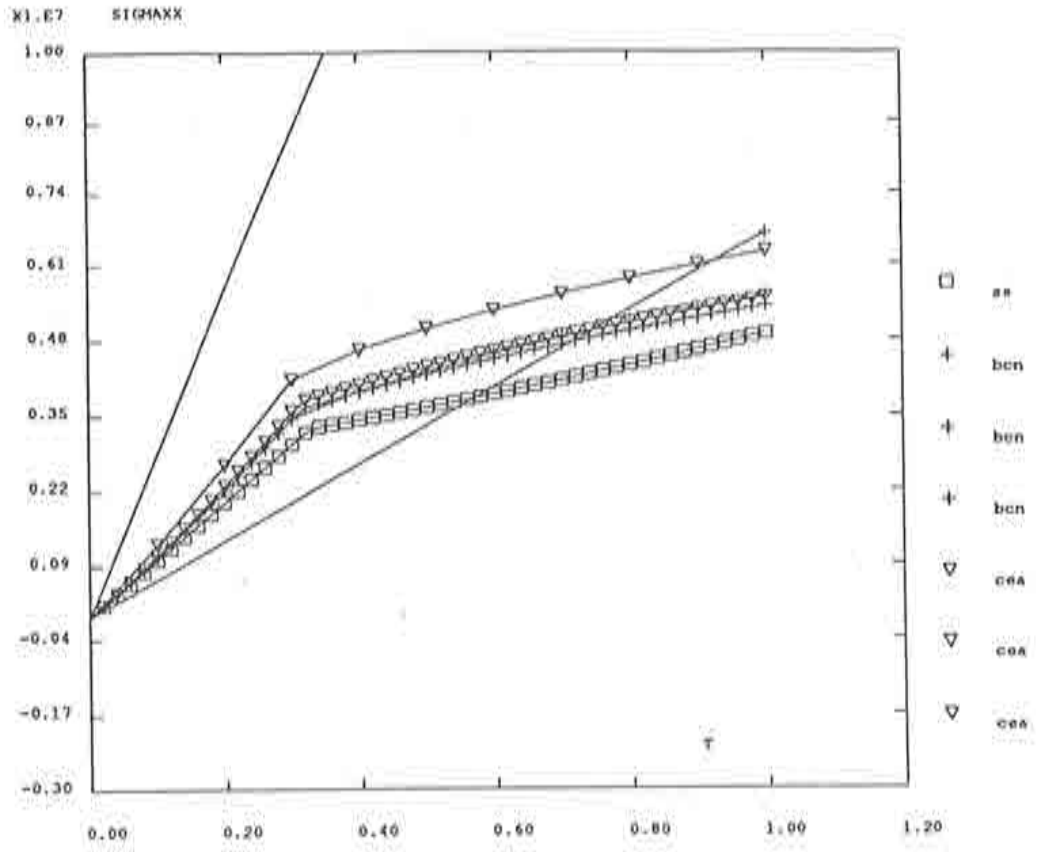


Figure 20: Extension and compression test, elastic analysis. Final value of stress ($t = 1$) vs. number of time steps (1, 2, 3, 5, 10, 20, 50). a) σ_{xx}. b) σ_{yy}.

a)



b)

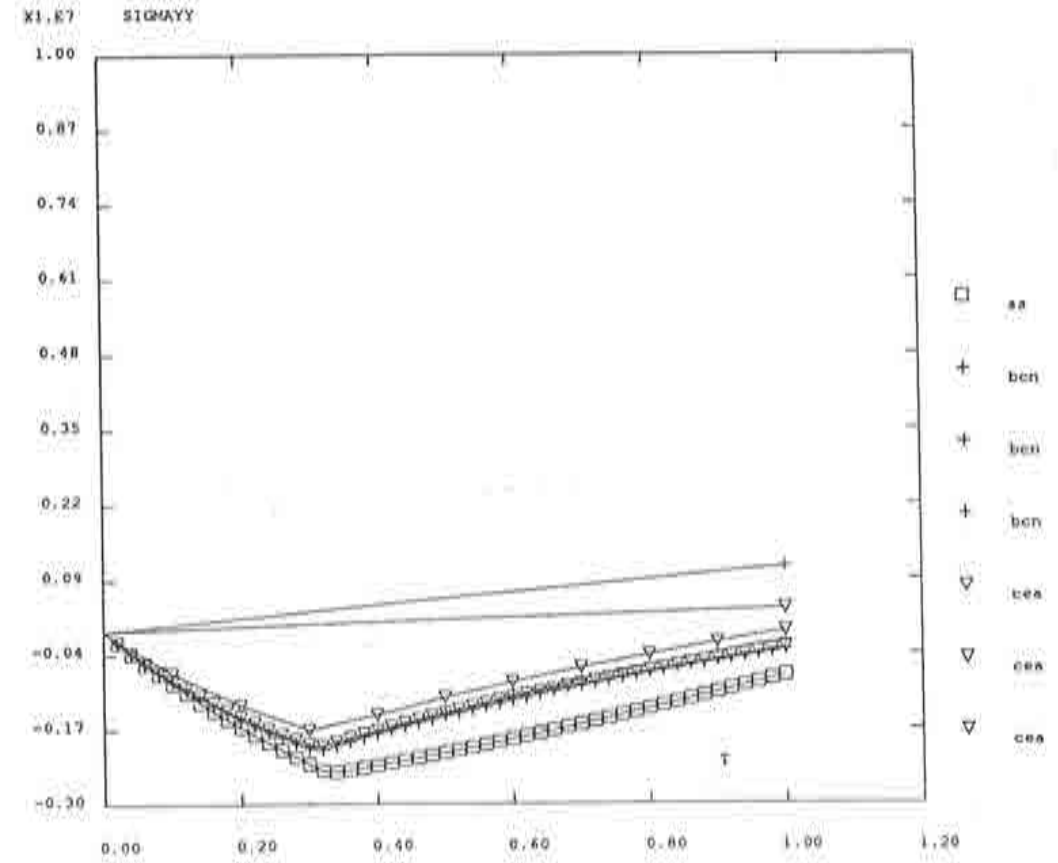


Figure 21: Extension and compression test, elastoplastic analysis. Stress vs. time curves computed with algorithms CEA and BCN (1, 10 and 50 time steps) and SS (small strain, 50 time steps). a) σ_{xx}. b) σ_{yy}.

4.5 Dilatation

The last deformation path corresponds to dilatation: a unit square undergoes biaxial extension, see Figure 22. The equations of motion are:

$$x(t) = X\,(1+t) \tag{67}$$

$$y(t) = Y\,(1+t)$$

resulting in the deformation gradient

$$F = \begin{bmatrix} 1+t & 0 \\ 0 & 1+t \end{bmatrix} \tag{68}$$

and the analytical solution to Eq. (45) is

$$\sigma_{xx}(t) = E\ \ln(1+t) \tag{69}$$

$$\sigma_{xy}(t) = 0$$

$$\sigma_{yy}(t) = E\ \ln(1+t)$$

As in the two previous tests, both CEA and BCN provide qualitatively correct results, in the sense that σ_{xy} is zero and σ_{xx} equals σ_{yy} for any number of time-steps and in both elastic and plastic modes. There are sharp differences, however, concerning convergence behaviour. For the elastic case, for instance, BCN provides a better prediction with one time increment than CEA with five, see Figures 23 and 24. A similar comparison is valid in the plastic test, where one BCN increment gets closer to the reference solution than ten CEA steps, see Figure 25.

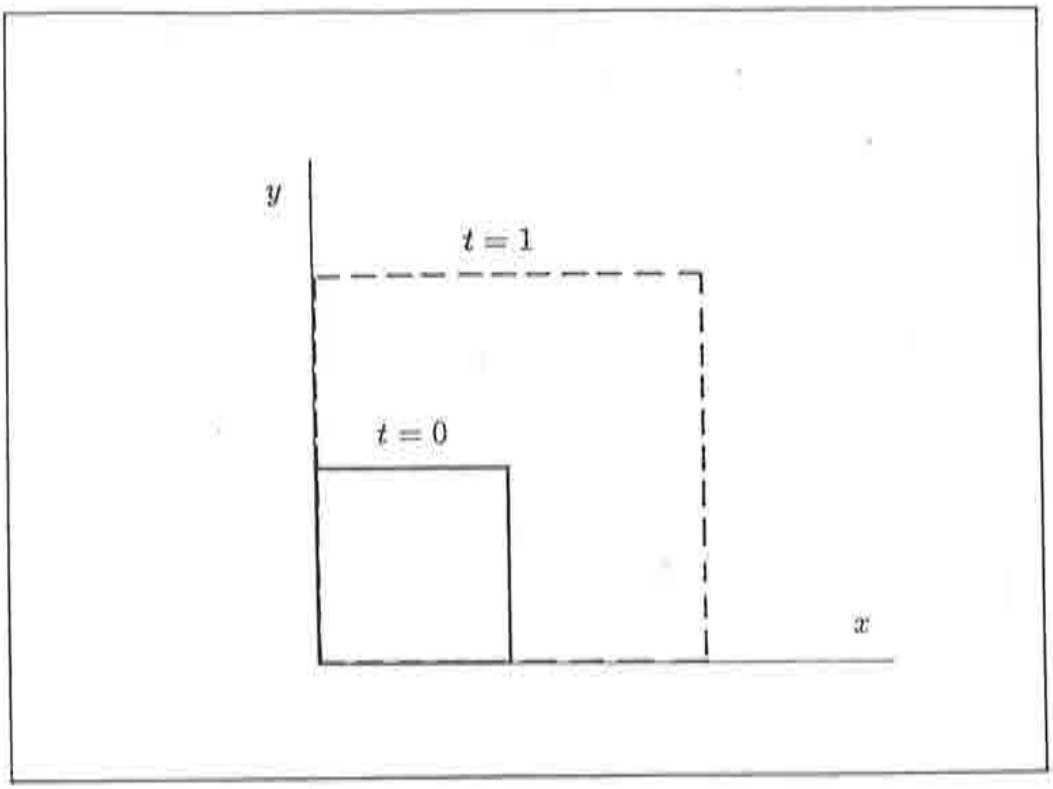


Figure 22: Dilatation test. Initial and final configurations.

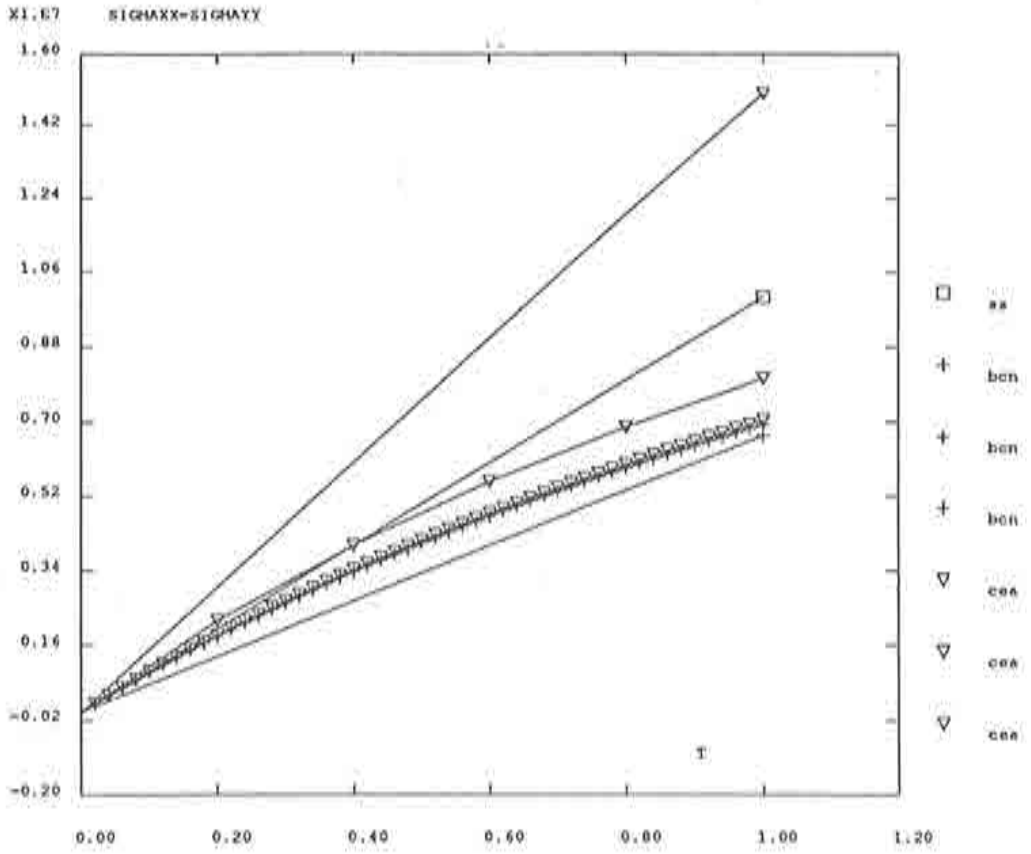


Figure 23: Dilatation test, elastic analysis. Normal stress vs. time curves computed with algorithms CEA and BCN (1, 5 and 50 time steps) and SS (small strain, 1 time step).

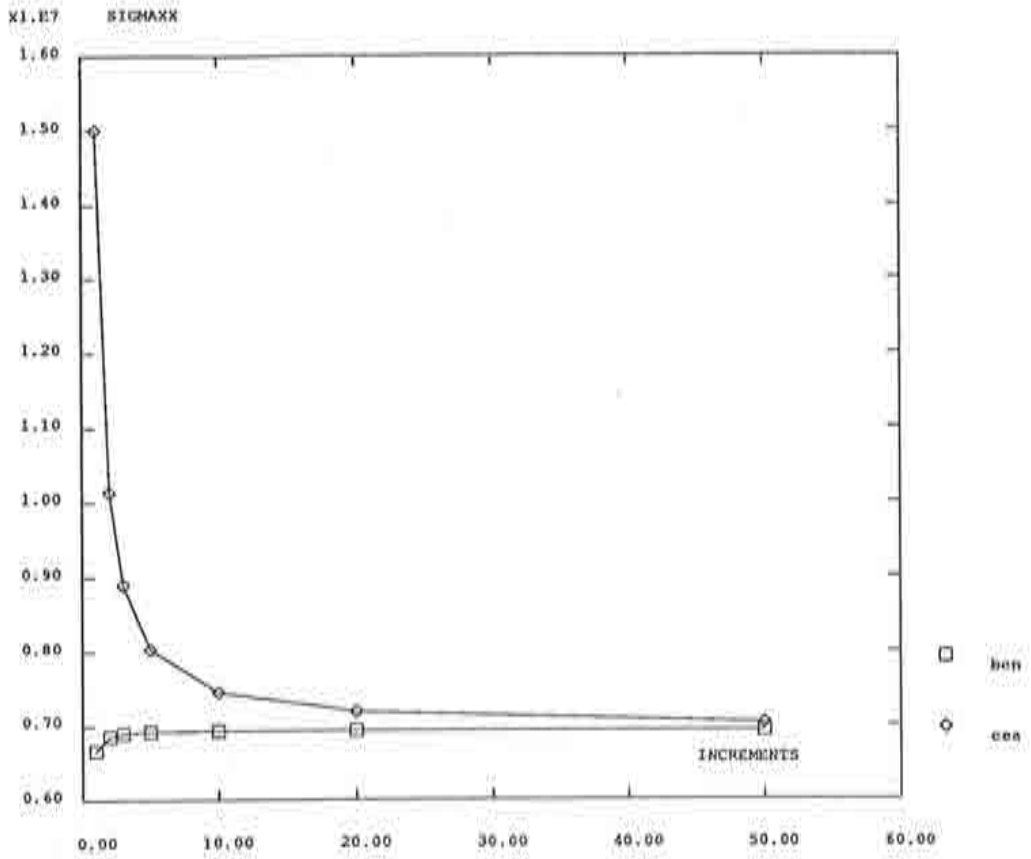


Figure 24: Dilatation test, elastic analysis. Final value of normal stress ($t = 1$) vs. number of time steps (1, 2, 3, 5, 10, 20, 50).

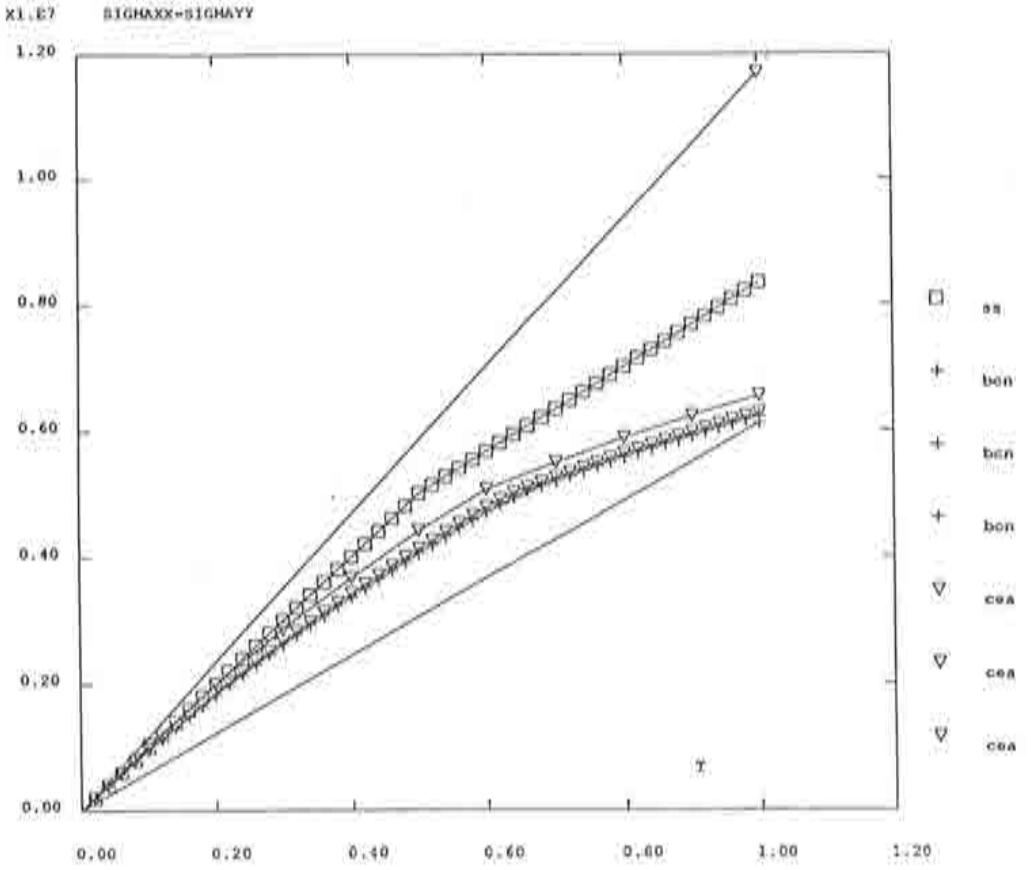


Figure 25: Dilatation test, elastoplastic analysis. Normal stress vs. time curves computed with algorithms CEA and BCN (1, 10 and 50 time steps) and SS (small strain, 50 time steps).

4.6 Benchmark test: necking of a circular bar

The necking problem is a well-known benchmark test in nonlinear solid mechanics, [19]. A circular bar, with a radius of 6.413 mm and 53.334 mm length , is subjected to uniaxial tension. A slight geometric imperfection (1% radius reduction), see Figure 26, induces necking in the central part of the bar. Because of axysimmetry, only a quarter of the specimen is modelled . The uniaxial constitutive law, [19], is presented in Figure 27.

A mesh of 50 eight-node quadrilaterals with 2x2 Gauss points is employed to simulate an axial elongation of 14 mm (26% of initial length). To account for the time-increment sensitivity of each algorithm, the computation is performed with three different time increments (100, 200, and 1000 steps) for both CEA and BCN.

The final deformed shape of the specimen is shown, for the reference solution (BCN with 1000 time steps), in Figure 28. The influence of time-increment can be seen in Figure 29, where radius reduction vs. elongation curves are displayed. The differences are important for the CEA algorithm, see Figure 29a, starting at around 10% elongation and leading to significantly different final values of the neck radius (21%) . On the other hand, BCN shows much less sensitivity to time-increment: the three curves, see Figure 29b, are much closer together, with the 200 and 1000 time-step solutions almost superimposed and a discrepancy of final neck radius of only 3%, and, moreover, are very similar to those portrayed in [19].

5. CONCLUDING REMARKS

Some basic notions on large strain solid mechanics have been reviewed, with particular emphasis on the principle of objectivity that constitutive equations must comply with: they must remain invariant under a time-dependent change of the reference frame. That means that only objective quantities may be employed and, since the material stress rate is not objective, alternative objective stress rates are needed. Some of the classical objective rates are presented and discussed.

A basic requirement on a numerical algorithm for the time-integration of the constitutive equations is that of *incremental objectivity*: if the strain state does not change during a time increment, the algorithm should provide a null variation of stress. Incremental objectivity is often viewed as the discrete counterpart of the notion of objectivity. However, since a rigid body motion during the increment is *assumed* as the most likely motion, we prefer the notion of *priority on rigid rotation motion* to the concept of *incremental objectivity*.

Two numerical algorithms that treat rigid rotations correctly have been discussed. The first one (CEA) is a generalization to large strains of the incremental form algorithm commonly employed in small strain analyses. The second one (BCN) is based on Truesdell objective stress rate.

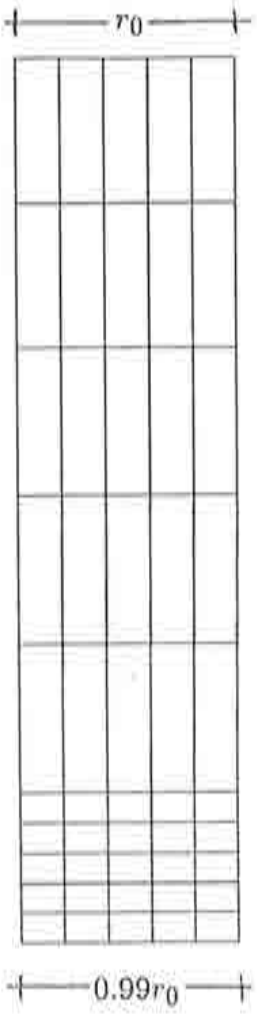


Figure 26: Necking problem.
Initial configuration and mesh.

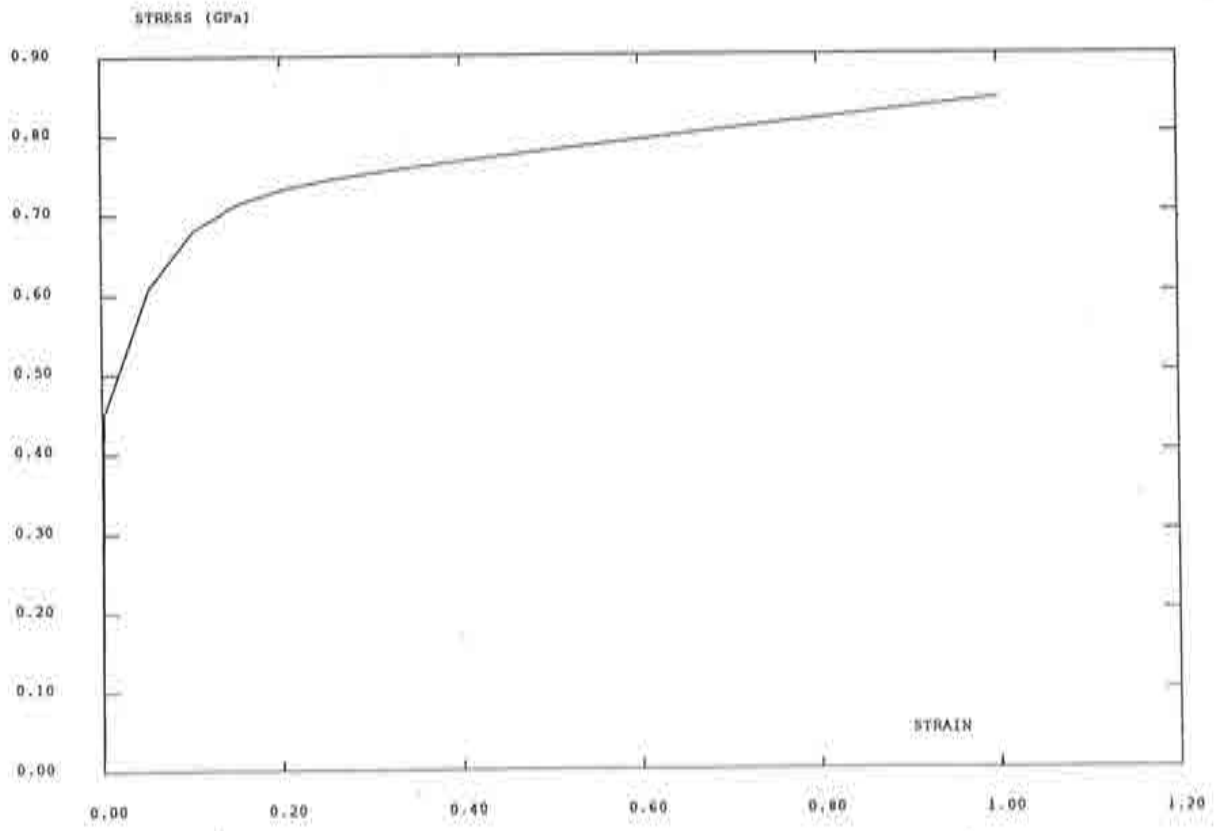


Figure 27: Elastoplastic constitutive equation for necking analysis.

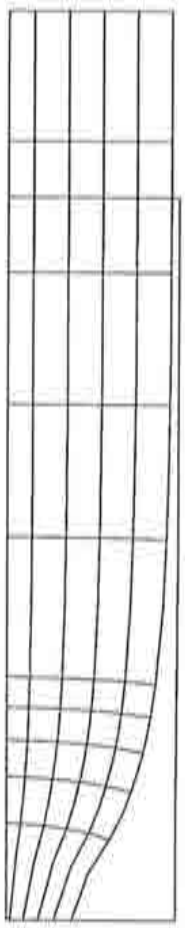


Figure 28: Final deformed shape. Algorithm BCN with 1000 time steps.

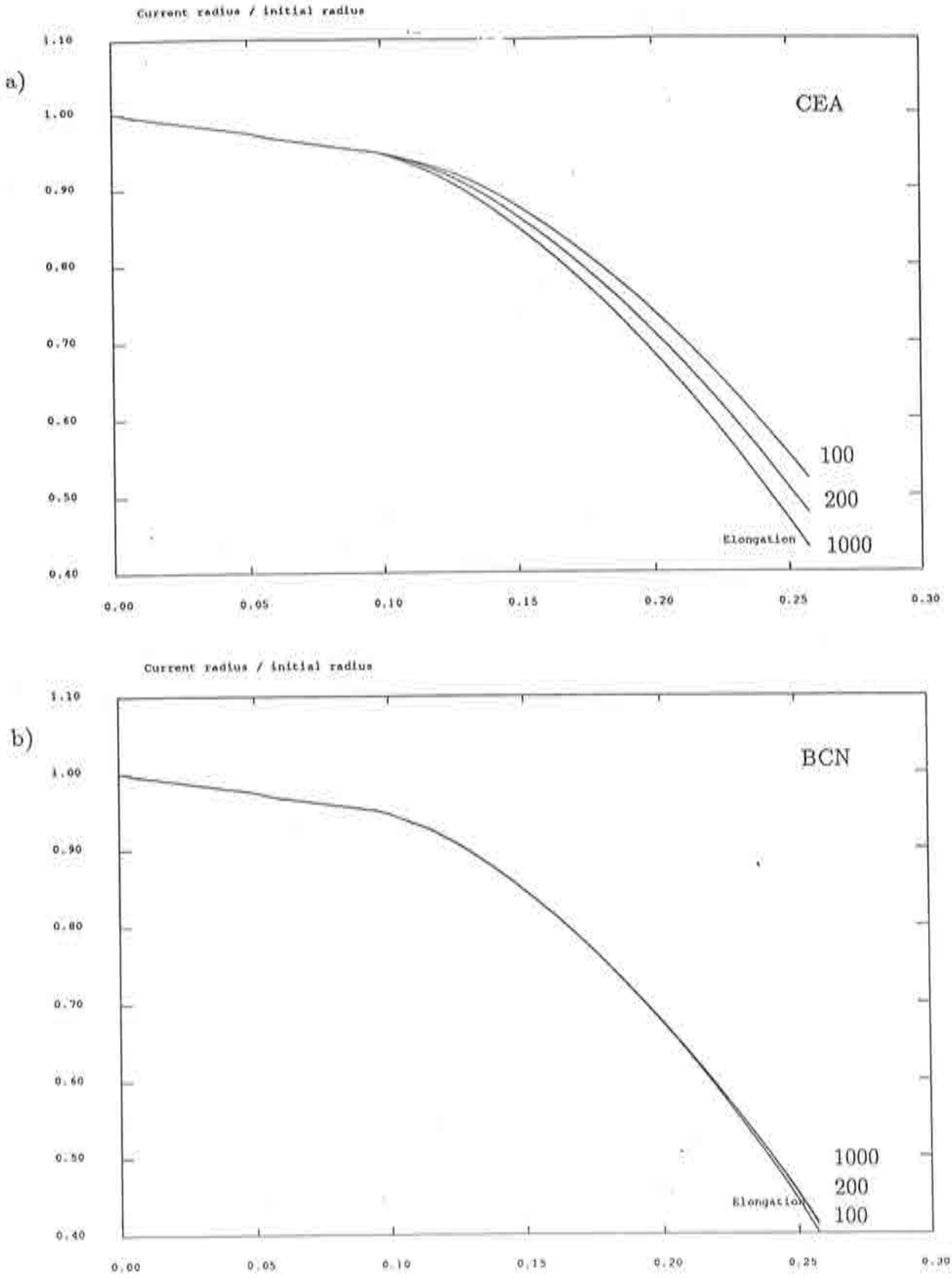


Figure 29: Radius reduction vs. bar elongation curves computed with 100, 200 and 1000 time steps. a) CEA. b) BCN.

These two algorithms have been compared with the aid of set of simple deformation paths (rigid rotation, simple shear, uniaxial extension, extension and compression, dilatation) and a necking analysis, a classical benchmark test in computational mechanics. The results show that algorithm BCN is consistently more accurate that algorithm CEA, the main reason being the use of the midstep configuration as a reference.

All the numerical tests have been performed in an object-oriented code, CASTEM2000. These codes store and manipulate information as objects which are far more complex than those in conventional, function-oriented codes. Operators are employed to create new objects from already existing objects. In this manner, an elementary process of a FEM computation is represented by a sentence of the meta-language of objects and operators provided by the code. The practical implications are of great importance: *i)* either when developing a new algorithm or when performing a test, the user/programmer thinks in terms of entities inherent to the FEM (mesh, stiffness matrix, model, material, nodal displacement field, ...) rather than in variables and arrays, as in a conventional code (matrix of nodal coordinates, connectivity matrix, vector of nodal displacements, ...). It has been shown here that performing a FEM analysis (Section 1.2) or developing a new algorithm (Annex 2) involves only a few lines in the meta-language offered by the object-oriented code.

ACKNOWLEDGEMENTS

The authors are indebted to the graduate student A. Vila for carrying out the computations of the various numerical tests. The authors also want to express their gratitude and appreciation to Dr. A. Millard for his interesting comments on the topic of large strains, and his help with CASTEM's time-integration algorithm.

ANNEX 1: stress update algorithm CEA

The stress update algorithm CEA is implemented in CASTEM2000 with the following lines in the GIBIANE language (within procedure INCREME):

```
1.)   sigi1 = PICA zmodl sigi0 zdept ;

2.)   dsig0 = SIGMA II zmodl zmat zdept ;

3.)   dsig1 = PICA zmodl dsig0 zdept ;

4.)   sigf1 = sigi1 + dsig1 ;
```

where the following objects and operators are involved:

`sigi0`: object of type MCHAML, initial stresses referred to initial configuration

`sigi1`: object of type MCHAML, initial stresses referred to final configuration

`dsig0`: object of type MCHAML, stress increment referred to initial configuration

`dsig1`: object of type MCHAML, stress increment referred to final configuration

`sigf1`: object of type MCHAML, final stresses referred to final configuration

`zmodl`: object of type MMODEL, model

`zmat`: object of type MCHAML, material properties

`zdept`: object of type CHPOINT, displacement increment between initial and final configurations

`PICA`: operator of forward Piola transformation. The second Piola-Kirchhoff stress tensor is pushed forward into the Cauchy stress tensor

`SIGMA II`: this operator computes the stresses corresponding to a given displacement field. The option II is employed to take into account the second-order terms in the evaluation of strains

The stress update can then be seen as consisting of four steps: 1.) the initial stress is pushed forward into the final configuration ; 2.) the stress increment is computed in the initial configuration by means of the full incremental Lagrange stress tensor and 3.) then pushed forward. Finally, 4.) the final stress is computed by adding initial stress and stress increment, both referred to the final configuration.

ANNEX 2: stress update algorithm BCN

The stress update algorithm BCN has been implemented in CASTEM2000 with the following lines in the GIBIANE language:

```
1.)  sigi1 = PICA zmod1 sigi0 zdept ;
2.)  zdep0in = alfa*zdept ; zdepin1 = (1-alfa)*zdept ;
3.)  FORM zdep0in ;
4.)  dsigin = SIGMA zmod1 zmat zdept ;
5.)  dsig1 = PICA zmod1 dsigin zdepin1 ;
6.)  sigf1 = sigi1 + dsig1 ;
```

Some of the objects and operators have already been presented in Annex 1. The new ones are:

`alfa`: object of type FLOTTANT, interpolation parameter for intermediate configuration

`zdep0in`: object of type CHPOINT, displacement increment between initial and intermediate configurations

`zdepin1`: object of type CHPOINT, displacement increment between intermediate and final configurations

`dsigin`: object of type MCHAML, stress increment referred to intermediate configuration

`FORM`: this operator updates the mesh geometry by means of a given displacement field

As in the algorithm of Annex 1, 1.) initial stresses are transported to final configuration. Then 2.) the total displacement increment is divided into two parts, to and from the intermediate configuration. After 3.) updating the mesh geometry, 4.) the stress increment is computed via operator `SIGMA` without option `II`, to account only for first-order terms. This stress increment is 5.) pushed forward into the final configuration and 6.) added to the initial stress.

REFERENCES

[1] Miller, G.R. (1991), "An Object-Oriented Approach to Structural Analysis and Design", *Computers & Structures*, Vol. 40, No. 1, pp. 75-82.

[2] Dubois-Pèlerin, Y., Zimmermann, Th. and Bomme, P. (1992), "Object-Oriented Finite Element Programming Comcepts", in New Advances in Computational Structural Mechanics, Elsevier, pp. 457-466.

[3] Mackie, R.I. (1992), "Object-Oriented Programming of the FEM", *International Journal for Numerical Methods in Engineering*, Vol. 35, No. 2, pp 425-436.

[4] Verpeaux, P. and Millard, A. (1988), "De l'existence à l'essence. De CASTEM à CASTEM2000. Quelques considérations sur le développement de grans codes du calcul", Rapport 88/179, Laboratoire d'Analyse Mécanique des Structures, C.E.A.

[5] Soria, A., Pegon, P., "Some introductory remarks about CASTEM2000", Personal communication, Applied Mechanics Division, Joint Research Centre, Ispra.

[6] "CASTEM2000 – Manuel d'utilisation", Rapport 88/176, Laboratoire d'Analyse Mécanique des Structures, C.E.A.

[7] Bretones, M.A., Rodríguez-Ferran, A. and Huerta, A. (1993) "Programación orientada al objeto: una herramienta de ingeniería y desarrollo", Proceedings of the Segundo Congreso de Métodos Numéricos en Ingeniería, La Coruña, Spain, pp. 273-282.

[8] Bathe, K.J. (1982), "Finite Element Procedures in Engineering Analysis", Prentice Hall, New Jersey, USA.

[9] "Modelling of Metal Forming Processes", edited by J.L. Chenot and E. Oñate, Kluwer Academic Publishers, 1988

[10] "Numerical Methods in Forming Processes", Special Issue of the *International Journal for Numerical Methods in Engineering*, Vol. 30, No. 8, December 1990.

[11] Pinsky, P.M., Ortiz, M. and Pister, K.S. (1983), "Numerical Integration of Rate Constitutive Equations in Finite Deformation Analysis", *Computer Methods in Applied Mechanics and Engineering*, Vol. 40, pp. 137-158.

[12] Prakash, N. (1981), "Differential Geometry – An Integrated Approach", Mc Graw-Hill, New Delhi, India.

[13] Marsden, J.E. and Hughes, T.J.R. (1983), " Mathematical Foundations of Elasticity", Prentice Hall, USA.

[14] Crisfield, M.A. (1991), "Non-linear Finite Element Analysis of Solids and Structures", John Wiley & Sons Ltd., England.

[15] Ortega, J.M., and Rheinboldt, W.C. (1970), "Iterative Solution of Nonlinear Equations in Several Variables", Academic Press, California, USA.

[16] Dennis, J.E. and Moré, J.J. (1977), "Quasi-Newton Methods: Motivation and Theory", *SIAM Review*, Vol. 19, pp. 46-89.

[17] Hughes, T.J.R. (1983), " Numerical Implementation of Constitutive Models: Rate-Independent Deviatoric Plasticity", Workshop on the Theoretical Foundation for Large-Scale Computations of Nonlinear Material Behavior, Northwestern Univ., Evanston, Illinois, USA.

[18] Pegon, P. (1993), Personal communication, Applied Mechanics Division, Joint Research Centre, Ispra.

[19] Simo, J.C. (1988), "A Framework for Finite Strain Elastoplasticity Based on Maximum Plastic Dissipation and the Multiplicative Decomposition. Part II: Computational Aspects", *Comp. Meths. Appl. Mech. Engrg.*, Vol. 68, pp. 1-31.